Liqing Zhang
Bao-Liang Lu
James Kwok (Eds.)

Advances in Neural Networks – ISNN 2010

LNCS 6063

7th International Symposium
on Neural Networks, ISNN 2010
Shanghai, China, June 2010, Proceedings, Part I

Springer

Lecture Notes in Computer Science 6063

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Liqing Zhang Bao-Liang Lu
James Kwok (Eds.)

Advances in Neural Networks – ISNN 2010

7th International Symposium
on Neural Networks, ISNN 2010
Shanghai, China, June 6-9, 2010
Proceedings, Part I

Springer

Volume Editors

Liqing Zhang
Bao-Liang Lu
Department of Computer Science and Engineering
Shanghai Jiao Tong University
800, Dongchuan Road
Shanghai 200240, China
E-mail: {zhang-lq; blu}@cs.sjtu.edu.cn

James Kwok
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong, China
E-mail: jamesk@cse.ust.hk

Library of Congress Control Number: Applied for

CR Subject Classification (1998): I.4, F.1, I.2, I.5, H.3, J.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-13277-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13277-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

This book and its sister volume collect refereed papers presented at the 7th International Symposium on Neural Networks (ISNN 2010), held in Shanghai, China, June 6-9, 2010. Building on the success of the previous six successive ISNN symposiums, ISNN has become a well-established series of popular and high-quality conferences on neural computation and its applications. ISNN aims at providing a platform for scientists, researchers, engineers, as well as students to gather together to present and discuss the latest progresses in neural networks, and applications in diverse areas. Nowadays, the field of neural networks has been fostered far beyond the traditional artificial neural networks.

This year, ISNN 2010 received 591 submissions from more than 40 countries and regions. Based on rigorous reviews, 170 papers were selected for publication in the proceedings. The papers collected in the proceedings cover a broad spectrum of fields, ranging from neurophysiological experiments, neural modeling to extensions and applications of neural networks. We have organized the papers into two volumes based on their topics. The first volume, entitled "Advances in Neural Networks-ISNN 2010, Part 1," covers the following topics: neurophysiological foundation, theory and models, learning and inference, neurodynamics. The second volume entitled "Advance in Neural Networks ISNN 2010, Part 2" covers the following five topics: SVM and kernel methods, vision and image, data mining and text analysis, BCI and brain imaging, and applications.

In addition to the contributed papers, four distinguished scholars (Andrzej Cichocki, Chin-Teng Lin, DeLiang Wang, Gary G. Yen) were invited to give plenary talks, providing us with the recent hot topics, latest developments and novel applications of neural networks.

ISNN 2010 was organized by Shanghai Jiao Tong University, Shanghai, China, The Chinese University of Hong Kong, China and Sponsorship was obtained from Shanghai Jiao Tong University and The Chinese University of Hong Kong. The symposium was also co-sponsored by the National Natural Science Foundation of China. We would like to acknowledge technical supports from the IEEE Shanghai Section, International Neural Network Society, IEEE Computational Intelligence Society, Asia Pacific Neural Network Assembly, International Association for Mathematics and Computers in Simulation, and European Neural Network Society.

We would like to express our sincere gratitude to the members of the Advisory Committee, Organizing Committee and Program Committee, in particular to Jun Wang and Zhigang Zeng, to the reviewers and the organizers of special sessions for their contributions during the preparation of this conference. We would like to also acknowledge the invited speakers for their valuable plenary talks in the conference.

Acknowledgement is also given to Springer for the continuous support and fruitful collaboration from the first ISNN to this seventh one.

March 2010

Liqing Zhang
James Kwok
Bao-Liang Lu

ISNN 2010 Organization

ISNN 2010 was organized and sponsored by Shanghai Jiao Tong University, The Chinese University of Hong Kong, and it was technically cosponsored by the IEEE Shanghai Section, International Neural Network Society, IEEE Computational Intelligence Society, Asia Pacific Neural Network Assembly, International Association for Mathematics and Computers in Simulation, and European Neural Network Society. It was financially supported by the National Natural Science Foundation of China.

General Chairs

Jun Wang	Hong Kong, China
Bao-Liang Lu	Shanghai, China

Organizing Committee Chair

Jianbo Su	Shanghai, China

Program Committee Chairs

Liqing Zhang	Shanghai, China
Zhigang Zeng	Wuhan, China
James T.Y. Kwok	Hong Kong, China

Special Sessions Chairs

Si Wu	Shanghai, China
Qing Ma	Kyoto, Japan
Paul S. Pang	Auckland, New Zealand

Publications Chairs

Hongtao Lu	Shanghai, China
Yinling Wang	Shanghai, China
Wenlian Lu	Shanghai, China

Publicity Chairs

Bo Yuan	Shanghai, China
Xiaolin Hu	Beijing, China
Qingshan Liu	Nanjing, China

Finance Chairs

Xinping Guan	Shanghai, China
Xiangyang Zhu	Shanghai, China

Registration Chairs

Fang Li	Shanghai, China
Gui-Rong Xue	Shanghai, China
Daniel W.C. Ho	Hong Kong, China

Local Arrangements Chairs

Qingsheng Ren	Shanghai, China
Xiaodong Gu	Shanghai, China

Advisory Committee Chairs

Xiaowei Tang	Hangzhou, China
Bo Zhang	Beijing, China
Aike Guo	Shanghai, China

Advisory Committee Members

Cesare Alippi, Milan, Italy
Shun-ichi Amari, Tokyo, Japan
Zheng Bao, Xi'an, China
Dimitri P. Bertsekas, Cabridge, MA, USA
Tianyou Chai, Shenyang, China
Guanrong Chen, Hong Kong
Andrzej Cichocki, Tokyo, Japan
Ruwei Dai, Beijing, China
Jay Farrell, Riverside, CA, USA
Chunbo Feng, Nanjing, China
Russell Eberhart, Indianapolis, IN, USA
David Fogel, San Diego, CA, USA
Walter J. Freeman, Berkeley, CA, USA
Kunihiko Fukushima, Osaka, Japan
Xingui He, Beijing, China
Zhenya He, Nanjing, China
Janusz Kacprzyk, Warsaw, Poland
Nikola Kasabov, Auckland, New Zealand
Okyay Kaynak, Istanbul, Turkey
Anthony Kuh, Honolulu, HI, USA
Frank L. Lewis, Fort Worth, TX, USA
Deyi Li, Beijing, China
Yanda Li, Beijing, China
Chin-Teng Lin, Hsinchu, Taiwan
Robert J. Marks II, Waco, TX, USA
Erkki Oja, Helsinki, Finland
Nikhil R. Pal, Calcutta, India
Marios M. Polycarpou, Nicosia, Cyprus
José C. Príncipe, Gainesville, FL, USA
Leszek Rutkowski, Czestochowa, Poland
Jennie Si, Tempe, AZ, USA
Youxian Sun, Hangzhou, China
DeLiang Wang, Columbus, OH, USA
Fei-Yue Wang, Beijing, China
Shoujue Wang, Beijing, China
Paul J. Werbos, Washington, DC, USA
Cheng Wu, Beijing, China
Donald C. Wunsch II, Rolla, MO, USA
Youlun Xiong, Wuhan, China

Lei Xu, Hong Kong
Shuzi Yang, Wuhan, China
Xin Yao, Birmingham, UK
Gary G. Yen, Stillwater, OK, USA
Nanning Zheng, Xi'an, China
Yongchuan Zhang, Wuhan, China
Jacek M. Zurada, Louisville, KY, USA

Program Committee Members

Haydar Akca
Alma Y. Alanis
Bruno Apolloni
Sabri Arik
Vijayan Asari
Tao Ban
Peter Baranyi
Salim Bouzerdoum
Martin Brown
Xindi Cai
Jianting Cao
Yu Cao
Jonathan Chan
Chu-Song Chen
Liang Chen
Sheng Chen
Songcan Chen
YangQuan Chen
Yen-Wei Chen
Zengqiang Chen
Jianlin Cheng
Li Cheng
Long Cheng
Zheru Chi
Sung-Bae Cho
Emilio Corchado
Jose Alfredo F. Costa
Ruxandra Liana Costea
Sergio Cruces
Baotong Cui
Chuanyin Dang
Mingcong Deng
Ming Dong
Jixiang Du
Andries Engelbrecht
Meng Joo Er
Jufu Feng
Chaojin Fu
Wai-Keung Fung
John Gan
Junbin Gao
Xiao-Zhi Gao
Xinping Guan
Chen Guo
Chengan Guo
Ping Guo
Abdenour Hadid
Honggui Han
Qing-Long Han
Haibo He
Hanlin He
Zhaoshui He
Akira Hirose
Daniel Ho
Noriyasu Homma
Zhongsheng Hou
Chun-Fei Hsu
Huosheng Hu
Jinglu Hu
Junhao Hu
Sanqing Hu
Guang-Bin Huang
Tingwen Huang
Wei Hui
Amir Hussain
Jayadeva
Minghui Jiang
Tianzi Jiang
Yaochu Jin
Joarder Kamruzzaman

Shunshoku Kanae
Qi Kang
Nik Kasabov
Okyay Kaynak
Rhee Man Kil
Kwang-Baek Kim
Sungshin Kim
Mario Koeppen
Rakhesh Singh Kshetrimayum
Edmund Lai
Heung Fai Lam
Minho Lee
Chi-Sing Leung
Henry Leung
Chuandong Li
Fang Li
Guang Li
Kang Li
Li Li
Shaoyuan Li
Shutao Li
Xiaoli Li
Xiaoou Li
Xuelong Li
Yangmin Li
Yuanqing Li
Yun Li
Zhong Li
Jinling Liang
Ming Liang
Pei-Ji Liang
Yanchun Liang
Li-Zhi Liao
Wudai Liao
Longnian Lin
Guoping Liu
Ju Liu
Meiqin Liu
Yan Liu
Hongtao Lu
Jianquan Lu
Jinhu Lu
Wenlian Lu
Jian Cheng Lv
Jinwen Ma
Malik Magdon Ismail
Danilo Mandic
Tiemin Mei
Dan Meng
Yan Meng
Duoqian Miao
Martin Middendorf
Valeri Mladenov
Marco Antonio Moreno-Armendáriz
Ikuko Nishkawa
Stanislaw Osowski
Seiichi Ozawa
Shaoning Pang
Jaakko Peltonen
Vir V. Phoha
Branimir Reljin
Qingsheng Ren
Tomasz Rutkowski
Sattar B. Sadkhan
Toshimichi Saito
Gerald Schaefer
Furao Shen
Daming Shi
Hideaki Shimazaki
Michael Small
Qiankun Song
Jochen J. Steil
John Sum
Roberto Tagliaferri
Norikazu Takahashi
Ah-hwee Tan
Ying Tan
Toshihisa Tanaka
Dacheng Tao
Ruck Thawonmas
Xin Tian
Christos Tjortjis
Ivor Tsang
Masao Utiyama
Marc Vanhulle
Bin Wang
Dan Wang
Dianhui Wang
Lei Wang
Liang Wang
Rubin Wang
Wenjia Wang
Wenwu Wang
Xiaoping Wang

Xin Wang
Yinglin Wang
Yiwen Wang
Zhanzhan Wang
Zhongsheng Wang
Zidong Wang
Hau-San Wong
Kevin Wong
Wei Wu
Cheng Xiang
Hong Xie
Songyun Xie
Rui Xu
Xin Xu
Guirong Xue
Yang Yang
Yingjie Yang
Yongqing Yang
Jianqiang Yi
Dingli Yu
Jian Yu
Xiao-Hua Yu
Bo Yuan
Kun Yuan
Pong C Yuen
Xiaoqin Zeng
Changshui Zhang
Jie Zhang
Junping Zhang
Kai Zhang
Lei Zhang
Nian Zhang
Dongbin Zhao
Hai Zhao
Liang Zhao
Qibin Zhao
Mingjun Zhong
Weihang Zhu

Reviewers

Ajith Abraham
Alma Y. Alanis
N.G. Alex
Jing An
Sung Jun An
Claudia Angelini
Nancy Arana-Daniel
Nancy Arana-Daniel
Kiran Balagani
Tao Ban
Simone Bassis
Anna Belardinelli
Joao Roberto Bertini Junior
Amit Bhaya
Shuhui Bi
Xuhui Bo
Salim Bouzerdoum
N. Bu
Qiao Cai
Xindi Cai
Hongfei Cao
Yuan Cao
Jonathan Chan
Wenge Chang
Benhui Chen
Bo-Chiuan Chen
Chao-Jung Chen
Chu-Song Chen
Cunbao Chen
Fei Chen
Gang Chen
Guici Chen
Junfei Chen
Lei Chen
Min Chen
Pin-Cheng Chen
Sheng Chen
Shuwei Chen
Tao Chen
Xiaofen Chen
Xiaofeng Chen
Yanhua Chen
Yao Chen
Zengqiang Chen
Zhihao Chen
Jianlin Cheng
K. H. Cheng
Lei Cheng
Yu Cheng
Yuhu Cheng
Seong-Pyo Cheon
Zheru Chi
Seungjin Choi
Angelo Ciaramella
Matthew Conforth
Paul Christopher Conilione
Paleologu Constantin
Jose Alfredo F. Costa
Ruxandra Liana Costea
Fangshu Cui
Zhihua Cui
James Curry
Qun Dai
Xinyu Dai
Spiros Denaxas
Jing Deng
Xin Deng
Zhijian Diao
Ke Ding
Jan Dolinsky

Yongsheng Dong
Adriao Duarte Doria Neto
Dajun Du
Jun Du
Shengzhi Du
Wei Du
Qiguo Duan
Zhansheng Duan
Julian Eggert
Yong Fan
Chonglun Fang
Italia De Feis
G.C. Feng
Qinrong Feng
Simone Fiori
Chaojin Fu
Jun Fu
Zhengyong Fu
Zhernyong Fu
Sheng Gan
Shenghua Gao
Fei Ge
Vanessa Goh
Dawei Gong
Weifeng Gu
Wenfei Gu
Renchu Guan
Chengan Guo
Jianmei Guo
Jun Guo
Ping Guo
Xin Guo
Yi Guo
Juan Carlos Gutierrez Caceres
Osamu Hasegawa
Aurelien Hazart
Hanlin He
Huiguang He
Lianghua He
Lin He
Wangli He
Xiangnan He
Zhaoshui He
Sc Ramon Hernandez Esteban Hernandez-Vargas
Kevin Ho
Xia Hong
Chenping Hou
Hui-Huang Hsu
Enliang Hu
Jinglu Hu
Junhao Hu
Meng Hu
Sanqing Hu
Tianjiang Hu
Xiaolin Hu
Zhaohui Hu
Bonan Huang
Chun-Rong Huang
Dan Huang
J. Huang
Kaizhu Huang
Shujian Huang
Xiaodi Huang
Xiaolin Huang
Zhenkun Huang
Cong Hui
GuoTao Hui
Khan M. Iftekharuddin
Tasadduq Imam
Teijiro Isokawa
Mingjun Ji
Zheng Ji
Aimin Jiang
Changan Jiang
Feng Jiang
Lihua Jiang
Xinwei Jiang
Gang Jin
Ning Jin
Yaochu Jin
Krzysztof Siwek
Yiannis Kanellopoulos
Enam Karim
Jia Ke
Salman Khan
Sung Shin Kim
Tae-Hyung Kim
Mitsunaga Kinjo
Arto Klami
Mario Koeppen
Adam Kong
Hui Kong
Qi Kong
Adam Krzyzak
Jayanta Kumar Debnath
Kandarpa Kumar Sarma
Franz Kurfess
Paul Kwan
Darong Lai
Jiajun Lai
Jianhuang Lai
Wei Lai
Heung Fai Lam
Paul Lam
Yuan Lan
Ngai-Fong Law
N. K. Lee
Chi SingLeung
Bing Li
Boyang Li
C. Li
Chaojie Li
Chuandong Li
Dazi Li
Guang Li
Junhua Li
Kang Li
Kelin Li
Li Li
Liping Li
Lulu Li
Manli Li
Peng Li
Ping Li
Ruijiang Li
Tianrui Li
Tieshan Li
Xiaochen Li
Xiaocheng Li
Xuelong Li
Yan Li
Yun Li
Yunxia Li
Zhenguo Li
Allan Liang
Jinling Liang
Pei-Ji Liang
Li-Zhi Liao

Wudai Liao
Hongfei Lin
Qing Lin
Tran Hoai Lin
Bo Liu
Chang Liu
Chao Liu
Fei Liu
Hongbo Liu
Jindong Liu
Lei Liu
Lingqiao Liu
Nianjun Liu
Qingshan Liu
Wei Liu
Xiangyang Liu
Xiwei Liu
Yan Liu
Yanjun Liu
Yu Liu
Zhaobing Liu
ZhenweiLiu
Jinyi Long
Jinyi Long
Carlos Lopez-Franco
Shengqiang Lou
Mingyu Lu
Ning Lu
S.F. Lu
Bei Lv
Jun Lv
Fali Ma
Libo Ma
Singo Mabu
Danilo Mandic
Qi Mao
Tomasz Markiewicz
Radoslaw Mazur
Tiemin Mei
Bo Meng
Zhaohui Meng
Marna van der Merwe
Martin Middendorf
N. Mitianoudis
Valeri Mladenov
Alex Moopenn
Marco Moreno
Loredana Murino
Francesco Napolitano
Ikuko Nishkawa
Tohru Nitta
Qiu Niu
Qun Niu
Chakarida Nukoolkit
Sang-Hoon Oh
Floriberto Ortiz
Stanislaw Osowski
Antonio de Padua Braga
Antonio Paiva
Shaoning Pang
Woon Jeung Park
Juuso Parkkinen
Michael Paul
Anne Magály de Paula Canuto
Zheng Pei
Jaakko Peltonen
Ce Peng
Hanchuan Peng
Jau-Woei Perng
Son Lam Phung
Xiong Ping
Kriengkrai Porkaew
Santitham Prom-on
Dianwei Qian
Lishan Qiao
Keyun Qin
Meikang Qiu
Li Qu
Marcos G. Quiles
Mihai Rebican
Luis J. Ricalde
Jorge Rivera
Haijun Rong
Zhihai Rong
Tomasz Rutkowski
Jose A. Ruz
Edgar N. Sanchez
Sergio P. Santos
Renato José Sassi
Chunwei Seah
NarimanSepehri
Caifeng Shan
Shiguang Shan
Chunhua Shen
Furao Shen
Jun Shen
Yi Shen
Jiuh-Biing Sheu
Licheng Shi
Qinfeng Shi
Xiaohu Shi
Si Si
Leandro Augusto da Silva
Angela Slavova
Sunantha Sodsee
Dandan Song
Dongjin Song
Doo Heon Song
Mingli Song
Qiang Song
Qiankun Song
Kingkarn Sookhanaphibarn
Gustavo Fontoura de Souza
Antonino Staiano
Jochen Steil
Pui-Fai Sum
Jian Sun
Jian-Tao Sun
Junfeng Sun
Liang Sun
Liming Sun
Ning Sun
Yi Sun
Shigeru Takano
Mingkui Tan
Ke Tang
Kecheng Tang
Y. Tang
Liang Tao
Yin Tao
Sarwar Tapan
Ruck Thawonmas
Tuan Hue Thi
Le Tian
Fok Hing Chi Tivive
Christos Tjortjis
Rutkowski Tomasz
Julio Tovar

Jianjun Tu
Zhengwen Tu
Goergi Tzenov
Lorenzo Valerio
Rodrigo Verschae
Liang Wan
Min Wan
Aihui Wang
Bin Wang
Bo Hyun Wang
Chao Wang
Chengyou Wang
Dianhui Wang
Guanjun Wang
Haixian Wang
Hongyan Wang
Huidong Wang
Huiwei Wang
Jingguo Wang
Jinghua Wang
Lan Wang
Li Wang
Lili Wang
Lizhi Wang
Min Wang
Ming Wang
Pei Wang
Ruizhi Wang
Xiaolin Wang
Xiaowei Wang
Xin Wang
Xu Wang
Yang Wang
Ying Wang
You Wang
Yunyun Wang
Zhanshan Wang
Zhengxia Wang
Zhenxing Wang
Zhongsheng Wang
Bunthit Watanapa
Hua-Liang Wei
Qinglai Wei
Shengjun Wen
Young-Woon Woo
Ailong Wu
Chunguo Wu
Jun Wu
Qiang Wu
Si Wu
Xiangjun Wu
Yili Xia
Zeyang Xia
Cheng Xiang
Linying Xiang
Shiming Xiang
Xiaoliang Xie
Ping Xiong
Zhihua Xiong
Fang Xu
Feifei Xu
Heming Xu
Jie Xu
LinLi Xu
Rui Xu
Weihong Xu
XianyunXu
Xin Xu
Hui Xue
Jing Yang
Liu Yang
Qingshan Yang
Rongni Yang
Shangming Yang
Wen-Jie Yang
Wenlu Yang
Wenyun Yang
Xubing Yang
Yan Yang
Yongqing Yang
Zi-Jiang Yang
John Yao
Jun Yao
Yingtao Yao
Keiji Yasuda
Ming-Feng Yeh
Xiao Yi
Chenkun Yin
Kaori Yoshida
WenwuYu
Xiao-Hua Yu
Kun Yuan
Weisu Yuan
Xiaofang Yuan
Zhuzhi Yuan
Zhuzhu Yuan
P.C. Yuen
Masahiro Yukawa
Lianyin Zhai
Biao Zhang
Changshui Zhang
Chen Zhang
Dapeng Zhang
Jason Zhang
Jian Zhang
Jianbao Zhang
Jianhai Zhang
Jianhua Zhang
Jin Zhang
Junqi Zhang
Junying Zhang
Kai Zhang
Leihong Zhang
Liming Zhang
Nengsheng Zhang
Nian Zhang
Pu-Ming Zhang
Qing Zhang
Shaohong Zhang
Tao Zhang
Teng-Fei Zhang
Ting Zhang
Xian-Ming Zhang
Yuyang Zhang
Hai Zhao
Qibin Zhao
Xiaoyu Zhao
Yi Zhao
Yongping Zhao
Yongqing Zhao
Ziyang Zhen
Chengde Zheng
Lihong Zheng
Yuhua Zheng
Caiming Zhong
Mingjun Zhong
Shuiming Zhong
Bo Zhou
Jun Zhou
Luping Zhou
Rong Zhou

Xiuling Zhou
Haojin Zhu
Song Zhu
Wenjun Zhu
Xunlin Zhu
Yuanming Zhu
Wei-Wen Zou
Xin Zou
Pavel Zuñiga

Secretariat

Jin Gang
Kan Hong
Qiang Wang
Qiang Wu
Rong Zhou
Tianqi Zhang

Table of Contents – Part I

Neurophysiological Foundation

Theory and Models

Learning and Inference

Neurodynamics

Table of Contents – Part II

SVM and Kernel Methods

Vision and Image

Data Mining and Text Analysis

BCI and Brain Imaging

Applications

Stimulus-Dependent Noise Facilitates Tracking Performances of Neuronal Networks

Longwen Huang[1] and Si Wu[2]

[1] Yuanpei Program and Center for Theoretical Biology, Peking University, Beijing, China
[2] Lab of Neural Information Processing, Institute of Neuroscience, Chinese Academy of Sciences, Shanghai, China

Abstract. Understanding why neural systems can process information extremely fast is a fundamental question in theoretical neuroscience. The present study investigates the effect of noise on speeding up neural computation. We consider a computational task in which a neuronal network tracks a time-varying stimulus. Two network models with varying recurrent structures are explored, namely, neurons have weak sparse connections and have strong balanced interactions. It turns out that when the input noise is Poissonian, i.e., the noise strength is proportional to the mean of the input, the network have the best tracking performances. This is due to two good properties in the transient dynamics of the network associated with the Poissonian noise, which are: 1) the instant firing rate of the network is proportional to the mean of the external input when the network is at a stationary state; and 2) the stationary state of the network is insensitive to the stimulus change. These two properties enable the network to track the stimulus change rapidly. Simulation results confirm our theoretical analysis.

Keywords: Neural Computation, Stochastic Noise, Transient Dynamics, Tracking Speed, Balanced Network and Fokker-Planck equation.

1 Introduction

Neural systems can process information extremely fast. Taking the visual system of primates as an example, event-related potential study has revealed that human subjects are able to carry out some complex scenes analysis in less than 150 ms [1]. Neurophysiological recording showed that the latency of neural response can be as short as 40 ms in V1 [2], and $80 - 110$ ms in the temporal cortex [3]. Understanding why neural systems can perform computation in such a rapid speed is of critical importance in our understanding the computational mechanisms of brain functions.

Recent studies on the dynamics of neuronal populations have suggested that stochastic noise, which is observed ubiquitously in biological systems and is often thought to degrade information processing, may actually play a critical role in speeding up neural computation [4,5]. The idea is intuitively understandable. In

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 1–8, 2010.

a noiseless environment, the speed of neural computation is limited by the membrane time constant of single neurons (in the order of $10-20$ ms). On the other hand, when inputs to a neural ensemble contain noises, noises can randomize the state of the network measured by the distribution of membrane potentials of all neurons. As a result, those neurons whose potentials are close to the threshold will fire rapidly after the onset of a stimulus, and conveys the stimulus information quickly to higher cortical areas. Although this computational picture has been widely recognized in the literature, there are some details concerning the performances of noises accelerating neural computation have not been well addressed, which particularly include: 1) the impact of noise structure, and 2) the impact of network topology, on the accelerating performance. The goal of this study is to investigate these two issues.

To demonstrate the computational speed of a network, we consider a tracking task in which the network tries to read-out a time-varying stimulus in time. We measure the discrepancy between the true stimulus values and the decoding results of the network. Two different noise forms, namely, the additive and the stimulus-dependent Gaussian white noises are compared.

2 The Models

The dynamics of a single neuron is modeled as an integrate-and-fire process, i.e.,

$$\tau \frac{dv_i}{dt} = -v_i + I_i(t), \tag{1}$$

where v_i represents the membrane potential of the ith neuron, τ the membrane time constant and $I_i(t)$ the synaptic current. A spike will be generated when the membrane potential of a neuron reaches a threshold θ, and immediately after firing, the membrane potential of the neuron is reset to be $v = 0$.

The synaptic current to a neuron is given by

$$I_i(t) = I_i^{rec}(t) + I_i^{ext}(t), \tag{2}$$

which consists of the recurrent input I_i^{rec} and the external input I_i^{ext}.

The recurrent input is given by

$$I_i^{rec}(t) = \sum_j w_{ij} \sum_m e^{-(t-t_j^m)/\tau_s}, \quad t_j^m \le t, \tag{3}$$

where w_{ij} is the connection weight from the jth neuron to the ith one, and τ_s the time constant of the synaptic current. t_j^m is the moment of the mth spike generated by the jth neuron. The form of w_{ij} is determined by the network topology.

The external input, which mimics the input current from other cortical or subcortical regions, is written as,

$$I_i^{ext}(t) = \mu + \sigma \xi_i(t), \tag{4}$$

where μ is the mean of the current and σ the noise strength. $\xi_i(t)$ is Gaussian white noise of zero mean and unit variance. The fluctuations of external inputs of different neurons are independent to each other, i.e., $< \xi_i(t_1)\xi_j(t_2) >= \delta_{ij}\delta(t_1 - t_2)$, where the symbol $< \cdot >$ denotes averaging over many trials.

We are interested in two noise forms, namely, the additive and the stimulus-dependent noises. For the additive one, the noise strength σ^2 is a constant and independent of μ. For the stimulus-dependent one, $\sigma^2 = \alpha\mu$. Note that when $\alpha = 1$, the noise is Poisson. We call the general case of $\alpha \neq 1$ the Poissonian noise.

We consider two different network models.

Model 1: Weak sparse recurrent interaction. The network consists of only excitatory neurons. Denote N the number of neurons, $N \gg 1$. In order to keep neurons fire irregularly and at low firing rates (to be biologically plausible), neuronal connections need to be sparse and random. We choose two neurons have a probability p to be connected, and p is small, e.g., we may choose $p = 0.1$, however, $Np \gg 1$ still holds. We set the weight $w_{ij} = 1/(Np)$ if there is a connection between neuron i and j, and $w_{ij} = 0$ otherwise. Thus, the total recurrent input to a neuron is in the order of one, and its fluctuations is in the order of $1/\sqrt{Np}$ and can be neglected.

Model 2: Strong balanced recurrent interaction. In a balanced network, neuronal connections are also sparse and random, however, the neuronal connection strength is much larger than that in Model 1. We set $w_{ij} \sim 1/\sqrt{NP}$. The total excitatory current to a neuron is then in the order of $\sqrt{NP}$, which needs to be balanced by inhibitory inputs, so that the overall recurrent input to a neuron is in the order of one. In the balanced network, the fluctuation of the overall recurrent input is in the order of one, which plays a critical role in driving the network dynamics.

3 Mean-Field Analysis

We apply mean-field approximation to analyze the population dynamics of two network models. For the convenience of analysis, we first consider there is no recurrent interaction between neurons and ignore the leakage term in the single neuron dynamics.

Denote $p(v,t)$ the distribution of membrane potentials of the neural ensemble. The Fokker-Planck equation for $p(v,t)$ is written as [6,7]

$$\tau \frac{\partial p(v,t)}{\partial t} = -\mu \frac{\partial p(v,t)}{\partial v} + \frac{\sigma^2}{2\tau} \frac{\partial^2 p(v,t)}{\partial t^2}. \tag{5}$$

The stationary distribution $p(v)$ of the network is calculated to be

$$p(v) = \begin{cases} \frac{1}{\theta}(1 - e^{-2\tau\theta/\beta})e^{2\tau v/\beta} & v < 0 \\ \frac{1}{\theta}(1 - e^{-2\tau(v-\theta)/\beta}) & 0 \leq v \leq \theta \\ 0 & v > \theta \end{cases} \tag{6}$$

where

$$\beta = \sigma^2/\mu \tag{7}$$

is only the parameter determining the shape of $p(v)$.

The firing rate of the network is calculated to be

$$r = \frac{\sigma^2}{2\tau^2}\frac{\partial p(v)}{\partial v}|_\theta = \frac{\mu}{\theta\tau}. \tag{8}$$

From the above results, we observe two interesting properties: 1) the mean of the external input μ is linearly encoded by the firing rate r of the network in the stationary state. This property is independent of the noise structure. 2) When the noise is Poissonian, i.e., $\sigma^2 = \alpha\mu$, the parameter $\beta = \alpha$, is a constant and independent of the input strength μ. This is critical for fast computation. It implies that the stationary distribution of membrane potentials of the network is invariant with respect to the change of external inputs.

3.1 Population Dynamics of Model 1

Denote r the firing rate of each neuron. With the mean-field approximation, we calculate the mean and the variance of recurrent input to a neuron, which are

$$\begin{aligned} < \sum_j w_{ij} \sum_m e^{-(t-t_j^m)/\tau_s} > &\approx Np\frac{1}{Np} < \int_{-\infty}^{t} e^{-(t-t')/\tau_s} dW > \\ &= r\tau_s, \end{aligned} \tag{9}$$

$$\begin{aligned} D(\sum_j w_{ij} \sum_m e^{-(t-t_j^m/\tau_s}) &= \frac{Np}{(Np)^2} D(\int_{-\infty}^{t} e^{-(t-t')/\tau_s} dW) \\ &\approx 0, \end{aligned} \tag{10}$$

where dW denotes a diffusion approximation of the Poisson process and the symbol $D(x)$ the variance of x.

Combining with the external input, the dynamics of a single neuron is written as,

$$\tau\frac{dv_i}{dt} = -v_i + (\mu + r\tau_s) + \sigma\xi_i. \tag{11}$$

Thus, under the mean-field approximation, the effect of the recurrent interaction is equivalent to changing the mean of the synaptic input to a neuron from μ to $\mu + r\tau_s$. Based on the above calculation, the stationary distribution of membrane potentials of the network is given by Eq.(6), and the corresponding shape parameter β and the network firing rate r_n are calculated to be

$$\beta = \frac{\sigma^2}{\mu + r\tau_s}, \tag{12}$$

$$r_n = \frac{\sigma^2}{2\tau^2}\frac{\partial p(v)}{\partial v}|_\theta = \frac{\mu + r\tau_s}{\theta\tau}. \tag{13}$$

In the stationary state, the network firing rate r_n (averaged over the neural population) equals to the firing rate r of individual neurons (averaged over time). From Eq.(13), it gives

$$r = \frac{\mu}{\theta\tau - \tau_s}, \tag{14}$$

and hence

$$\beta = \frac{(\theta\tau - \tau_s)\sigma^2}{\theta\tau\mu}. \tag{15}$$

Again, we observe two good properties: 1) the mean of the externa input is linearly encoded by the firing rate of the network in the stationary state; and 2) when the noise is Poissonian, $\beta = \alpha(\theta\tau - \tau_s)/(\theta\tau)$, the distribution of membrane potentials of the network is independent of the input strength μ.

3.2 Population Dynamics of Model 2

Denote N_E and N_I the numbers of excitatory and inhibitory neurons in the network, respectively, and $K_E = pN_E$ and $K_I = pN_I$ the average numbers of excitatory and inhibitory connections a neuron may receive. We set $w_{ij}^{EE} = J_E/\sqrt{K_E}$, and $w_{kl}^{IE} = J_E/\sqrt{K_E}$, with a probability p and zero otherwise, and set $w_{ij}^{II} = -J_I/\sqrt{K_I}$, and $w_{kl}^{EI} = -J_I/\sqrt{K_I}$ if two neurons have a connection (with a probability p) and zero otherwise. r_E and r_I represent the firing rates of excitatory and inhibitory neurons.

With the mean-field approximation, the mean and the variance of a neuron's recurrent inputs are calculated to be,

$$< \sum_j w_{ij}^{ab} \sum_m e^{-(t-t_{j,b}^m)/\tau_{b,s}} > = \sqrt{K_b} J_b r_b \tau_{b,s}, \tag{16}$$

$$D(\sum_j w_{ij}^{ab} \sum_m e^{-(t-t_{j,b}^m)/\tau_{b,s}}) = \frac{(J_b)^2 r_b \tau_{b,s}}{2}, \tag{17}$$

where the variables a and b represent E or I.

Combining with the external inputs, we have

$$\begin{aligned} \tau_E \frac{dv_{i,E}}{dt} &= -v_i + (\mu + \sqrt{K_E} J_E r_E \tau_{E,s} + \sqrt{K_I} J_I r_I \tau_{I,s}) \\ &\quad + \sqrt{\sigma^2 + \frac{(J_E)^2 r_E \tau_{E,s}}{2} + \frac{(J_I)^2 r_I \tau_{I,s}}{2}} \xi_i, \end{aligned} \tag{18}$$

$$\begin{aligned} \tau_I \frac{dv_{i,I}}{dt} &= -v_i + (\sqrt{K_E} J_E r_E \tau_{E,s} + \sqrt{K_I} J_I r_I \tau_{I,s}) \\ &\quad + \sqrt{\frac{(J_E)^2 r_E \tau_{E,s}}{2} + \frac{(J_I)^2 r_I \tau_{I,s}}{2}} \xi_i. \end{aligned} \tag{19}$$

Thus, under the mean-field approximation, the effect of recurrent interactions in the balanced network is equivalent to changing the mean and the variance of the synaptic input properly. Following the same calculations as in Model 1, the

stationary distributions of membrane potentials of the excitatory and inhibitory neuron pools satisfy the same distribution as in Eq.(6), except that the shape parameters β_E and β_I are changed accordingly, which are,

$$\beta_E = \frac{\sigma^2 + 0.5(J_E)^2 r_E \tau_{E,s} + 0.5(J_I)^2 r_I \tau_{I,s}}{\mu + \sqrt{K_E} J_E r_E \tau_{E,s} + \sqrt{K_I} J_I r_I \tau_{I,s}}, \tag{20}$$

$$\beta_I = \frac{(J_E)^2 r_E \tau_{E,s} + (J_I)^2 r_I \tau_{I,s}}{2\sqrt{K_E} J_E r_E \tau_{E,s} + 2\sqrt{K_I} J_I r_I \tau_{I,s}}. \tag{21}$$

The firing rate of each neuron pool, which equals to the firing rate of individual neurons in the stationary state, is calculated to be:

$$r_E = \frac{\mu + \sqrt{K_E} J_E r_E \tau_{E,s} + \sqrt{K_I} J_I r_I \tau_{I,s}}{\theta \tau_E}, \tag{22}$$

$$r_I = \frac{\sqrt{K_E} J_E r_E \tau_{E,s} + \sqrt{K_I} J_I r_I \tau_{I,s}}{\theta \tau_I} \tag{23}$$

By the self-consistent condition, we have

$$r_E = \frac{\theta \tau_I - J_I \sqrt{K_I} \tau_{I,s}}{\theta^2 \tau_E \tau_I - \theta \tau_E J_I \sqrt{K_I} \tau_{I,s} - \theta \tau_I J_E \sqrt{K_E} \tau_{E,s}} \mu \tag{24}$$

$$r_I = \frac{J_E \sqrt{K_E} \tau_{E,s}}{\theta^2 \tau_E \tau_I - \theta \tau_E J_I \sqrt{K_I} \tau_{I,s} - \theta \tau_I J_E \sqrt{K_E} \tau_{E,s}} \mu \tag{25}$$

Thus, in the balanced network, the mean of the external input is linearly encoded by the firing rate of the network.

When the noise is Poissonian, we get

$$\begin{aligned} \beta_E = & \frac{(J_E)^2 \theta \tau_{E,s} \tau_I - (J_E)^2 J_I \tau_{E,s} \tau_{I,s} \sqrt{K_I} + (J_I)^2 J_E \tau_{E,s} \tau_{I,s} \sqrt{K_E}}{2\theta^2 \tau_E \tau_I - 2\theta J_I \sqrt{K_I} \tau_{I,s} \tau_E} \\ & + \frac{\alpha(\theta \tau_E \tau_I - J_E \sqrt{K_E} \tau_{E,s} \tau_I - J_I \sqrt{K_I} \tau_{I,s} \tau_E)}{\theta \tau_E \tau_I - J_I \sqrt{K_I} \tau_{I,s} \tau_E}, \end{aligned} \tag{26}$$

$$\beta_I = \frac{J_E \theta \tau_{E,s} \tau_I - J_E J_I \tau_{E,s} \tau_{I,s} \sqrt{K_I} + (J_I)^2 \tau_{E,s} \tau_{I,s} \sqrt{K_E}}{2\sqrt{K_E} \tau_I \tau_{E,s} \theta}. \tag{27}$$

Both β_E and β_I are independent of the mean of the external input.

4 Tracking a Time-Varying Stimulus

In a noise environment, since individual neurons fire irregularly, it is the statistical properties of network response that encodes external stimuli. Furthermore, for fast computation, it is the transient dynamics of the network that conveys the stimulus information. The transient dynamics of a network is affected by the noise form and the initial state of the network. In order to achieve fast and

Table 1. The fitted parameters

	Model 1	Model 2
Poissonian Noise	$a = 0.83, \phi = -0.04$	$a = 0.94, \phi = -0.09$
Additive Noise	$a = 0.67, \phi = -0.42$	$a = 0.90, \phi = -0.20$

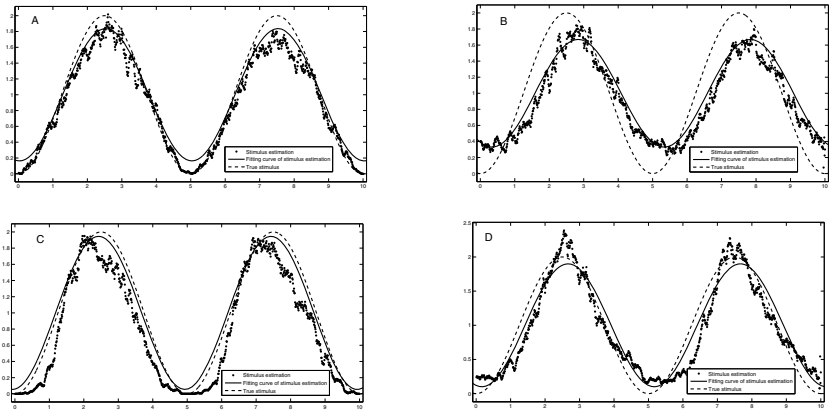


Fig. 1. Tracking performances of two network models. $T = 5\tau$.(A) Model 1 with the Poissonian noise; (B) Model 1 with the additive noise; (C) Model 2 with the Poissonian noise; and (D) Model 2 with the additive noise.

reliable computation, it is important that the statistical properties of the transient dynamics of a network is insensitive to input changes. So, what kind of noise structure is most suitable for fast neural computation, in particular, for the tracking task we consider?

In the above analysis, for two network models, we have found that when the input noise is Poissonian, the network transient dynamics has two important properties, which are: 1) the mean of external input is linearly encoded by the instant firing rate of the network when the network is at a stationary state; and 2) the stationary state of the network is insensitive to the change of the stimulus value (the mean of external input). These two good properties ensure that the Poissonian noise is ideal for fast tracking. In the fast tracking process, the stimulus value changes rapidly. The fact that the stationary state of the network is insensitive to the stimulus value implies that the network is always in a good state to catch up with the change; otherwise, the network has to evolve its state to a stationary one which is time-consuming.

We carry out simulation to confirm our theoretical analysis. We consider the mean of the external input changes with time, i.e, $\mu = 1 - cos(2\pi t/T)$. The parameter T controls the change speed of the stimulus. We measure firing rates of the network at different time, and fit them with a function $r = 1 - acos(2\pi t/T + \phi)$. The phase ϕ, which typically has a negative value, reflects the amount of delay in tracking. The deviation of a from the value one reflects the discrepancy

between the network decoding and the true stimulus. Apparently, the closer the value of ϕ to zero and the value of a to one, the better the tracking performance is.

Fig. 1 illustrates the tracking performances of two network models with $T = 5\tau$ and varied noise forms. The additive noise strength is set to be $\sigma^2 = 1$. The fitted parameters are summarized in Table 1. We see that for two network models, the tracking performances in the case of the Poissonian noise are better than that in the case of the additive noise.

5 Conclusions

The present study investigates the ideal noisy environment for fast neural computation. We observe that the stimulus-dependent Poissonian noise, rather than stimulus-independent ones, has the best effect of accelerating neural information processing. This property is also intuitively understandable. For the strong diffusive noise, in a short time-window, it is fluctuations, rather than the mean drift, that dominates the value of external inputs (that is, $W(t) \approx \mu t + \sigma\eta\sqrt{t} \approx \sigma\eta\sqrt{t}$, for $t \ll 1$, where η is a Gaussian random number of zero mean and unit variance). The signal-noise correlation is the key that enables the stimulus information to be adequately propagated to the neural system quickly.

References

1. Thorpe, S., Fize, D., Marlot, C.: Speed of Processing in the Human Visual System. Nature 381, 520–522 (1996)
2. Celebrini, S., Thorpe, S., Trotter, Y., Imbert, M.: Dynamics of Orientation Coding in Area V1 of the Awake Primate. Vis. Neurosci. 10, 811–825 (1993)
3. Sugase, Y., Yamane, S., Ueno, S., Kawano, K.: Global and Fine Information Coded by Single Neurons in the Temporal Visual Cortex. Nature 400, 869–873 (1999)
4. Gerstner, W.: Population Dynamics of Spiking Neurons: Fast Transients, Asynchronous State and Locking. Neural Computation 12, 43–89 (1999)
5. van Rossum, M., Turrigiano, G., Nelson, S.: Fast Propagation of Firing Rates through Layered Networks of Neurons. J. Neurosci. 22, 1956–1966 (2002)
6. Tuckwell, H.: Introduction to Theoretical Neurobiology. Cambridge University Press, Cambridge (1996)
7. Brunel, N., Hakim, V.: Fast Global Oscillations in Networks of Integarte-and-Fire Neurons with Low Firing Rates. Neural Computation 11, 1621–1671 (1999)

Range Parameter Induced Bifurcation in a Single Neuron Model with Delay-Dependent Parameters

Min Xiao[1,2] and Jinde Cao[2]

[1] School of Mathematics and Information Technology,
Nanjing Xiaozhuang University, Nanjing 210017, China
[2] Department of Mathematics, Southeast University, Nanjing 210096, China
candymanxm2003@yahoo.com.cn

Abstract. This paper deals with the single neuron model involving delay-dependent parameters proposed by Xu et al. [Phys. Lett. A, 354, 126-136, 2006]. The dynamics of this model are still largely undetermined, and in this paper, we perform some bifurcation analysis to the model. Unlike the article [Phys. Lett. A, 354, 126-136, 2006], where the delay is used as the bifurcation parameter, here we will use range parameter as bifurcation parameter. Based on the linear stability approach and bifurcation theory, sufficient conditions for the bifurcated periodic solution are derived, and critical values of Hopf bifurcation are assessed. The amplitude of oscillations always increases as the range parameter increases; the robustness of period against change in the range parameter occurs.

Keywords: Bifurcation, Oscillations, Delay-dependent parameters.

1 Introduction

For single neuron dynamics, Gopalsamy and Leung [1] proposed the following model of differential equation:

$$\dot{x}(t) = -x(t) + a\tanh[x(t)] - ab\tanh[x(t-\tau)]. \tag{1}$$

Here, $x(t)$ denotes the neuron response, and a and b are the range of the continuous variable $x(t)$ and measure of the inhibitory influence from the past history, respectively. Recently, Pakdaman and Malta [2], Ruan et al. [3] and Liao et al. [4] studied the stability, bifurcation and chaos of (1). But, the aforementioned studies on (1) suppose that the parameters in this model are constant independent of time delay. However, memory performance of the biological neuron usually depends on time history, and its memory intensity is usually lower and lower as time is gradually far away from the current time. It is natural to conceive that these neural networks may involve some delay-dependent parameters. Therefore, Xu et al. [5,6] considered (1) with parameter b depending on time delay τ described by

$$\dot{x}(t) = -\mu x(t) + a\tanh[x(t)] - ab(\tau)\tanh[x(t-\tau)], \tag{2}$$

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 9–16, 2010.

where $\mu > 0, a > 0, \tau \geq 0$ is the time delay and $b(\tau) > 0$, which is called memory function, is a strictly decreasing function of τ. The presence of such dependence often greatly complicates the task of an analytical study of such model. Most existing methods for studying bifurcation fail when applied to such a class of delay models. Compared with the intensive studies on the neural networks with delay-independent parameters, little progress has been achieved for the systems that have delay-dependent parameters. Although a detailed analysis on the stability switches, Hopf bifurcation and chaos of (2) with delay-dependent parameters is given in [5,6], the dynamics analysis of (2) is far from complete. The purpose of this paper is to perform a thorough bifurcation analysis on (2). Unlike in Xu et al. [5,6], where the delay τ is used as the bifurcation parameter, here we will use the range parameter a as bifurcation parameter. Based on the linear stability approach and bifurcation theory, critical values of Hopf bifurcation are assessed, and sufficient conditions for the bifurcated periodic solution are derived. Moreover, the amplitude of oscillations always increases as the range parameter increases; the robustness of period against change in the range parameter occurs.

2 Bifurcation from Range Parameter a

The linearization of (2) at $x = 0$ is

$$\dot{x}(t) = (-\mu + a)x(t) - ab(\tau)x(t - \tau), \tag{3}$$

whose characteristic equation is

$$\lambda = -\mu + a - ab(\tau)e^{-\lambda\tau}. \tag{4}$$

In what follows, we regard range parameter a as the bifurcation parameter to investigate the distribution of the roots to (4).

Lemma 1. *For each fixed $\tau > 0$, if $0 < a \leq \mu/(1 + b(\tau))$, then all the roots of (4) have negative real parts.*

Proof. When $a = 0, \lambda = -\mu < 0$. For $a > 0$, clear $\lambda = 0$ is not a root of (4) since $a(1-b(\tau)) < a(1+b(\tau)) \leq \mu$. Let $\mathrm{i}\omega(\omega > 0)$ be a root of (4), it is straightforward to obtain that

$$ab(\tau)\cos(\omega\tau) = a - \mu, \qquad ab(\tau)\sin(\omega\tau) = \omega, \tag{5}$$

yielding $\omega^2 = [ab(\tau)]^2 - (a-\mu)^2$. If $a \leq \mu/[1+b(\tau)]$ holds, we have $ab(\tau) \leq |\mu - a|$. Thus, (4) has no imaginary root. In other words, (4) has no root appearing on the imaginary axis for $a \in (0, \mu/(1 + b(\tau))]$. Recalling that the root of (4) with $a = 0$ has negative real part, the conclusion follows.

Lemma 2. *For each fixed $\tau > 0$, there exists a sequence of a, denoted as $a_j, j = 1, 2, \ldots$, such that (4) has a pair of purely imaginary roots $\pm i\omega_j$ when $a = a_j$; where*

$$a_j = \frac{\omega_j}{b(\tau)\sin(\omega_j\tau)}, \tag{6}$$

ω_j is the root of the equation

$$\omega \cot(\omega\tau) + \mu = \frac{\omega}{b(\tau)\sin(\omega\tau)}. \tag{7}$$

Proof. If $\lambda = \mathrm{i}\omega$ is a pure imaginary solution of (4), it must satisfy (5). Then, (7) can be directly from (5). Solutions of this equation are the horizontal coordinates of the intersecting points between the curve $y = \omega\cot(\omega\tau)$ and $y = \omega/[b(\tau)\sin(\omega\tau)] - \mu$. There are infinite number of intersecting points for these two curves that are graphically illustrated in Fig. 1. Denote ω_j as the solution of (7), and define a_j as in (6) , then (ω_j, a_j) is a solution of (5). Clearly, (4) has a pair of purely imaginary roots $\pm\mathrm{i}\omega_j$ when $a = a_j$. This completes the proof.

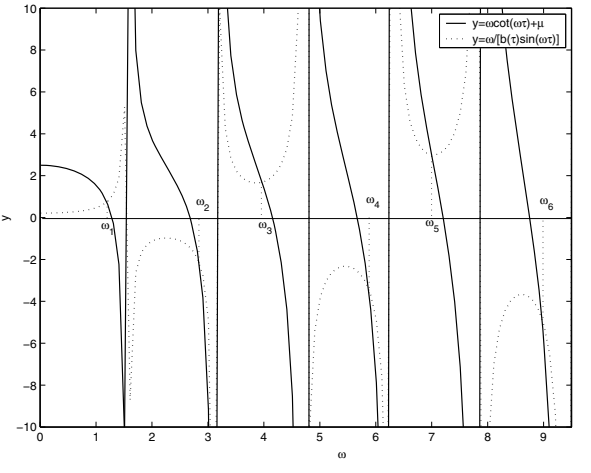


Fig. 1. Illustration for intersecting points between curve $y = \omega\cot(\omega\tau) + \mu$ and $y = \omega/[b(\tau)\sin(\omega\tau)]$

The last condition for the occurrence of a Hopf bifurcation at $a = a_j$ is

$$\frac{\mathrm{d}}{\mathrm{d}a}[\mathrm{Re}\lambda]_{a=a_j} \neq 0. \tag{8}$$

Substituting $\lambda(a)$ into (4) and differentiating the resulting equation with respect to a, we get

$$\frac{\mathrm{d}\lambda}{\mathrm{d}a} = \frac{\lambda + \mu}{a[1 + \tau(\lambda + \mu - a)]},$$

and hence,

$$\left.\frac{\mathrm{d}\lambda}{\mathrm{d}a}\right|_{a=a_j} = \frac{\mu[1 + \tau(\mu - a_j)] + \tau\omega_j^2 + (1 - \tau a_j)\omega_j\mathrm{i}}{a_j[(1 + \tau(\mu - a_j))^2 + \tau^2\omega_j^2]}.$$

Thus, we have

$$\frac{\mathrm{d}}{\mathrm{d}a}[\mathrm{Re}\lambda]_{a=a_j} = \frac{\mu[1+\tau(\mu-a_j)]+\tau\omega_j^2}{a_j[(1+\tau(\mu-a_j))^2+\tau^2\omega_j^2]}.$$

Theorem 1. *Suppose that* $\frac{d}{da}[Re\lambda]_{a=a_j} \neq 0$. *Then, for the system (2), there exists a Hopf bifurcation emerging from its trivial equilibrium* $x = 0$, *when the range parameter,* a, *passes through the critical value,* $a = a_j, j = 1, 2, \ldots$, *where* a_j *is defined by (6)-(7).*

Proof. The transversality condition (8) for Hopf bifurcation is satisfied. Applying Lemma 2 and Hopf bifurcation theorems for functional differential equations in [7], we obtain that Hopf bifurcation occurs at $a = a_j$ for the system (2). This completes the proof.

Without loss of generality, we only consider the intersecting points with positive horizontal coordinates $\omega_j, j = 1, 2, \ldots$, in Fig.1. It is clear that $\omega_1 < \omega_2 < \omega_3 < \cdots$, and $\omega_j \to \infty$ monotonically when $j \to \infty$. For these ω_j, form (6) and Fig.1, we obtain the following ordering

$$\cdots < a_6 < a_4 < a_2 < 0 < a_1 < a_3 < a_5 < \cdots.$$

Thus, there exists a minimum positive number a_1 such that (4) has a pair of purely imaginary roots $\pm i\omega_1$ at $a = a_1$. From Lemmas 1 and 2, we easily obtain the following results about the stability of the trivial equilibrium $x = 0$ of system (2).

Theorem 2. *Suppose that* $\frac{d}{da}[Re\lambda]_{a=a_1} > 0$. *Then, for each fixed* $\tau > 0$, *the trivial equilibrium* $x = 0$ *of system (2) is asymptotically stable when* $a \in (0, a_1)$, *and unstable when* $a > a_1$.

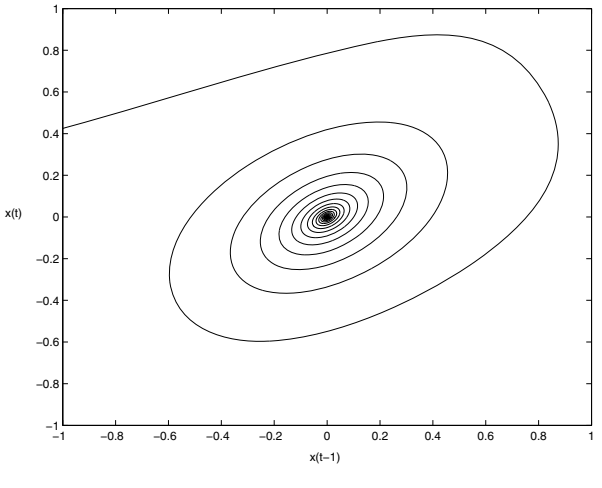


Fig. 2. Waveform plot and phase portrait of system (2) with $a = 0.68$

Proof. It is well known that the solution is locally asymptotically stable if all the roots of the characteristic equation have negative real parts and unstable if at least one root has positive real part. Therefore, conclusions is straightforward from Lemmas 1 and 2. This completes the proof.

3 Numerical Simulations

To verify the results obtained in the previous section, some examples are given as following. For comparison, the similar model (2), used in [5], is discussed.

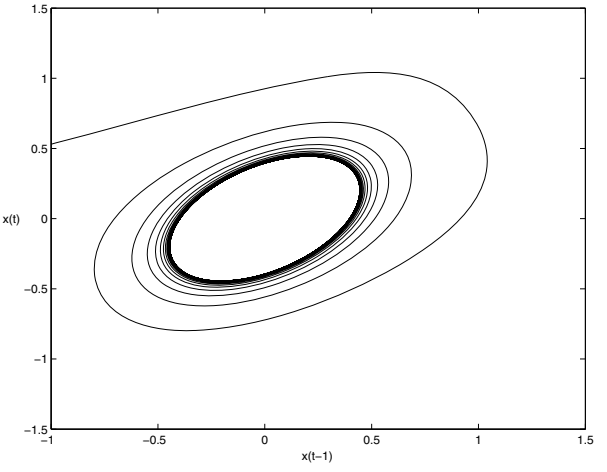


Fig. 3. Phase portrait of system (2) with $a = 0.77$

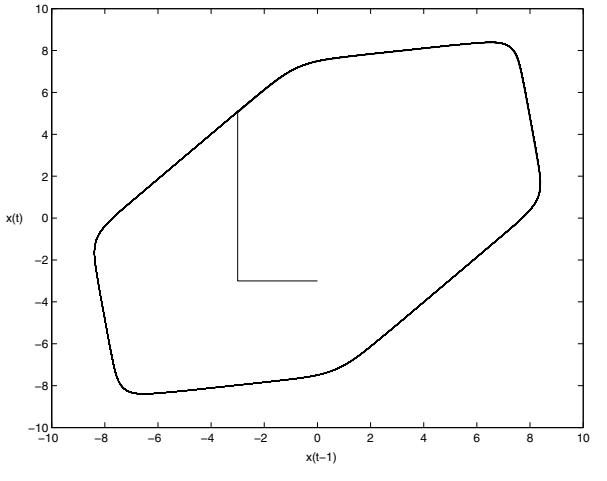


Fig. 4. Phase portrait of system (2) with $a = 5$

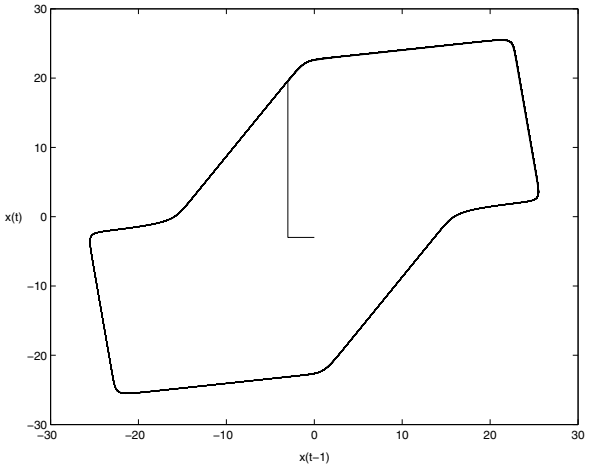


Fig. 5. Phase portrait of system (2) with $a = 15$

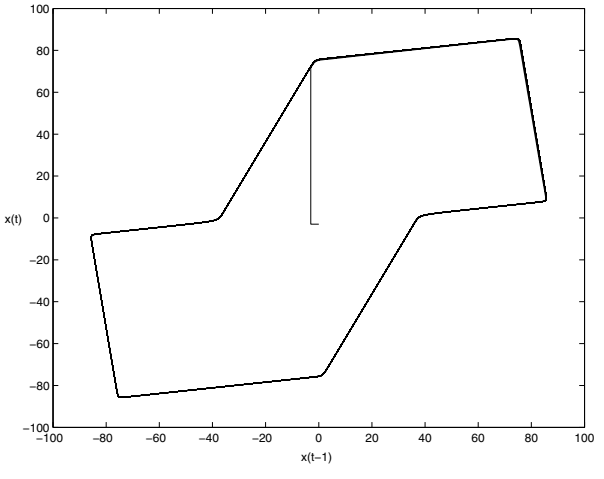


Fig. 6. Phase portrait of system (2) with $a = 50$

Table 1. Period and amplitude of the oscillations at different values of a

Range parameter a	0.77	5	15	50
Period	5.8	6.1	6.38	6.8
Amplitude	0.9	16.4	50	166

In our simulations of (2) with memory function $b(\tau) = be^{-\alpha\tau}(b > 0, \alpha > 0)$, $\mu = 2, \tau = 2, b = 3$ and $\alpha = 0.12$. We can apply (6)-(7) in Lemma 2 to obtain $a_1 = 0.7351$. From Theorem 2, we know that the trivial equilibrium $x = 0$ of system (2) is asymptotically stable when $a \in (0, 0.7351)$. This is illustrated

by the numerical simulation shown in Fig. 2 in which $a = 0.68$. Further, from Theorem 1, when a is increased to the critical value 0.7351, the trivial equilibrium $x = 0$ losses its stability and Hopf bifurcation occurs. The bifurcation is supercritical and the bifurcating periodic solution is asymptotically stable (see Figs. 3-6).

Table 1 shows the effect of the range parameter a on the oscillation period and amplitude. The amplitudes increase clearly with the range parameter a, which means that the amplitudes of the oscillations can be controlled by regulating range parameter a. The amplitude is always sensitive to the change of the range parameter a. In addition to amplitude, the period of oscillation remains around 6.5 when the range parameter a varies. The change of period is not sensitive to a. The robustness of period against change in the range paramete occurs. Figs. 3-6 show the sustained oscillations generated by (2) with different a.

4 Concluding Remarks

This paper deals with the dynamics of a neuron model with delay-dependent parameters, which is far from complete. Unlike in Xu et al. [5], where the delay is used as the bifurcation parameter, here we choose range parameter as bifurcation parameter. A series of critical range parameters are determined and a simple stable criterion is given according to the range parameter. Through the analysis for the bifurcation, it is shown that the trivial equilibrium may lose stability via a Hopf bifurcation. The amplitudes of oscillations always increase clearly as the range parameter increases. In addition, the robustness of period against change in range parameter occurs.

The range parameter play an important role in dynamical behaviors of neural network model (2) with delayed dependent parameters. We can control the dynamical behaviors of the model (2) by modulating the range parameter. The method proposed in this paper is important for understanding the regulatory mechanisms of neural network. Moreover, the method provides a control mechanism to ensure a transition from an equilibrium to a periodic oscillation with a desired and robust amplitude and period.

Acknowledgement. This work was jointly supported by the National Natural Science Foundation of China under Grant 60874088, the 333 Project of Jiangsu Province of China, and the Specialized Research Fund for the Doctoral Program of Higher Education under Grant 20070286003. This work was also jointly sponsored by the Qing Lan Project of Jiangsu Province, the China Postdoctoral Science Foundation funded project under Grant 20090461056, the Jiangsu Planned Projects for Postdoctoral Research Funds under Grant 0901025C, the Jiangsu Ordinary University Natural Science Research Project under Grant 09KJD110007 and the Fundamental Discipline Construction Foundation of Nanjing Xiaozhuang University.

References

1. Gopalsamy, K., Leung, I.: Convergence under Dynamical Thresholds with Delays. IEEE Trans. Neural Netw. 8, 341–348 (1997)
2. Pakdaman, K., Malta, C.P.: A Note on Convergence under Dynamical Thresholds with Delays. IEEE Trans. Neural Netw. 9, 231–233 (1998)
3. Ruan, J., Li, L., Lin, W.: Dynamics of Some Neural Network Models with Delay. Phys. Rev. E 63, 051906 (2001)
4. Liao, X.F., Wong, K.W., Leung, C.S., Wu, Z.F.: Hopf Bifurcation and Chaos in A Single Delayed Neuron Equation with Non-Monotonic Activation Function. Chaos, Solitons and Fractals 12, 1535–1547 (2001)
5. Xu, X., Hua, H.Y., Wang, H.L.: Stability Switches, Hopf Bifurcation and Chaos of A Neuron Model with Delay-Dependent Parameters. Phys. Lett. A 354, 126–136 (2006)
6. Xu, X., Liang, Y.C.: Stability and Bifurcation of A Neuron Model with Delay-Dependent Parameters. In: Wang, J., Liao, X.-F., Yi, Z. (eds.) ISNN 2005. LNCS, vol. 3496, pp. 334–339. Springer, Heidelberg (2005)
7. Hale, J.: Theory of Functional Differential Equations. Springer, New York (1977)

Messenger RNA Polyadenylation Site Recognition in Green Alga *Chlamydomonas Reinhardtii*

Guoli Ji[1,*], Xiaohui Wu[1,2], Qingshun Quinn Li[2], and Jianti Zheng[1]

[1] Department of Automation, Xiamen University, Xiamen 361005, China
[2] Department of Botany, Miami University, Oxford, Ohio 45056, USA

Abstract. Recognition of polyadenylation [poly(A)] sites for messenger RNA is important in genome annotation and gene expression regulation analysis. In the paper, poly(A) sites of *Chlamydomonas reinhardtii* were identified using an updated version of poly(A) site recognition software PASS_VAR based on generalized hidden Markov model. First, we analyzed the characteristics of the poly(A) sites and their surrounding sequence patterns, and used an entropy-based feature selection method to select important poly(A) signal patterns in conservative signal states. Then we improved the existing poly(A) sites recognition software PASS that was initially designed only for Arabidopsis to make it suitable for different species. Next, *Chlamydomonas* sequences were grouped according to their signal patterns and used to train the model parameters through mathematical statistics methods. Finally, poly(A) sites were identified using PASS_VAR. The efficacy of our model is showed up to 93% confidence with strong signals.

Keywords: Alga, Polyadenylation, GHMM, Site recognition, Entropy.

1 Introduction

The messenger RNA (mRNA) post-transcriptional processing includes three steps: the formation of 5' cap structure, the splicing of introns and the 3'-end polyadenylation at the untranslated region (UTR) [1]. The 3'-end polyadenylation is essential to the regulation of mRNA stability, mRNA intracellular transport and translation, as well as relating with a number of other mechanisms for cellular functions and disease responses [2, 3]. Accurate identification of the positions of polyadenylation [Poly(A) site] contributes to the determination of a gene's end and structures [4].

To predict poly(A) sites, at present there are methods such as Erpin [5] and Polyadq [6] which are applied to human genomes, and others such as PASS [7] and a classification-based prediction model [8] applied to Arabidopsis. However, currently available tools or methods are species specific, and in particular when introns (similar nucleotide distributions as 3'-UTR) interference is considered, the prediction results may not reflect the authentic poly(A) sites. In animals, the characteristic

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 17–26, 2010.

sequences around poly(A) sites are highly conservative, especially in mammals, the proportion of hexamer AATAAA is up to 80% [9], thus the accuracy of poly(A) sites identification of animal is relatively high [10]. (Here, to simplify sequence statement, we use T instead of U as in RNA sequence because we are dealing with DNA sequences.) In contrast, due to the fact that plant *cis*-element sequences are much less conserved (only about 10% genes having AATAAA) and with great variability [11], as well as the limit knowledge and lack of information about the *cis*-elements, the plant poly(A) sites identification is much more difficult.

Chlamydomonas reinhardtii is a green algal species that is widely used as a single eukaryotic cell model to study photosynthesis and cellular movements' mechanisms [12, 13]. More importantly, studies of this and other algae may lead to effective use of them for renewal energy production, e.g. biomass and diesel fuel [14]. In this paper, the poly(A) site recognition tool PASS [7], which was designed based on poly(A) signal profiles from Arabidopsis, was improved to be flexible for different species by setting different parameters. Then the characteristics of *Chlamydomonas* poly(A) signals and the surrounding *cis*-elements were analyzed to build recognition model based on Generalized Hidden Markov Model (GHMM) [15]. Next, the sequences were grouped according to the signal patterns and the model parameters of each group was trained. Especially for the NUE (near upstream element) signal state which is most conservative, the entropy based feature selection method was used for filtering optimal NUE patterns. Finally the model was used to predict *Chlamydomonas* ploy(A) sites and the experimental results showed its efficiency.

2 Data Sets

A dataset of 16,952 sequences with authentic poly(A) sites (referred to as 17k dataset) [16] were used. The length of each sequence was 400 nucleotides (or nt), and the known poly(A) site was in position 300. We also adopt several control data sets from JGI (*Chlamydomonasdomonas reinhardtii* Assembly v.4.0; Joint Genome Institute of the US Department of Energy, http://genome.jgi-psf.org). Sequences of 400nt to 1000nt were obtained, including 1658 5'-UTR, 17032 introns, 11471 CDS. And then 100 non-TGTAA containing sequences were randomly selected for testing. In addition, another control set of randomly Markov 1-order sequences was generated.

3 Poly(A) Site Recognition Model

3.1 Topology of the Recognition Model

The Generalized Hidden Markov Model (GHMM) [15] is used to identify poly(A) sites. The topology of the recognition model is shown in Fig. 1, where each state has different nucleotide output probabilities, and the state transition probability between each state is 1.

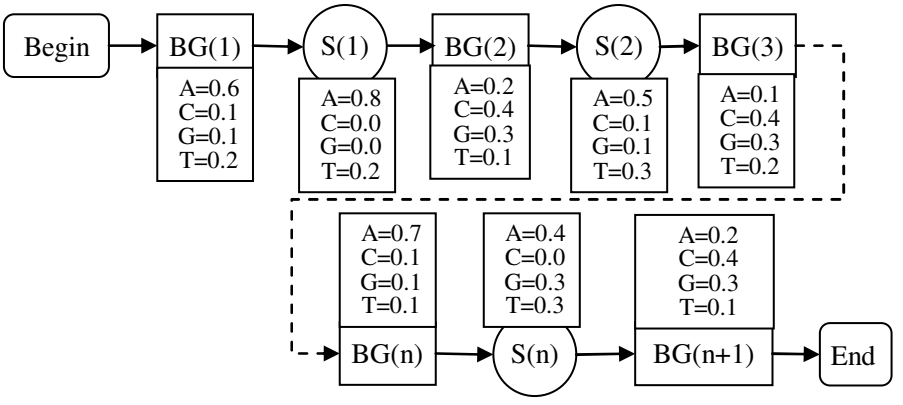


Fig. 1. Topology of the recognition model. Each node represents a state, S denotes the signal state, BG denotes the background state between two signal states.

3.2 Recognition Model and Forward-Backward Algorithm

The parameters of the recognition model are shown in Table 1. Given a nucleotide sequence $o = o_1 o_2 ... o_T$, each nucleotide of the sequence will correspond to a score using the identification model λ. The score is the possibility of the location being a poly(A) site, that is, the probability of generating the observation sequence $p(o \mid \lambda)$ which can be calculated by the forward-backward algorithm.

Table 1. Parameters of the recognition model

Parameter	Description
$S = \{S_1, S_2, \cdots, S_N\}$	Set of states. N is the number of the states.
V={ $v_1, v_2, \ldots, v_M$ }	Observation symbols. M is the number of observation symbols.
A={ a_{ij} }	The state transition matrix from state S_i to S_j
B={ $b_j(k)$ }	Output probability of each observation symbol in state S_j
π ={ π_i }	The initial state probability

First, define the partial probability $\alpha_t(i) = p(o_1 o_2 ... o_t, q_t = S_i \mid \lambda)$ in the forward algorithm as the probability of observation sequence at all times before and at t $o_1 o_2 ... o_t$ being at time t and in state S_i. Calculating steps of $\alpha_t(i)$ are as follows:

1) Initiation:

$$\alpha_t(i) = \pi_i b_i(o_1) \quad 1 \le \mathrm{i} \le \mathrm{N} \tag{1}$$

2) Recursion:

$$\alpha_{t+1}(j)=\left[\sum_{i=1}^{N}\alpha_t(i)a_{ij}\right]b_j(o_{t+1}) \quad 1\le \mathrm{t}\le \mathrm{T}-1,\ 1\le \mathrm{j}\le \mathrm{N} \tag{2}$$

3) Termination:

$$p(o\mid\lambda)=\sum_{i=1}^{N}\alpha_T(i) \tag{3}$$

Then, define the partial probability $\beta_t(i)=p(o_{t+1}o_{t+2}...o_T|\ q_t=S_i,\lambda)$ in the backward algorithm as the probability of observation sequence at all times after t+1 to T being at time t and in state S_i. Calculating steps of $\beta_t(i)$ are similar to those of $\alpha_t(i)$.

Finally, assume the poly(A) site is in state n, then the probability of the sequence o in position t in the state n (that is, poly(A) site) is $p_t(n,o\mid\lambda)=\alpha_{t-1}(n)\beta_t(n+1)$.

3.3 Improved Poly(A) Sites Recognition Software – PASS_VAR

A poly(A) site recognition software called Poly(A) Site Sleuth (PASS) [7] was developed based on GHMM. PASS is mainly suitable for the identification the poly(A) sites of Arabidopsis, but also has been used for rice whose nucleotide distribution is very similar to that of Arabidopsis [17]. However, the poly(A) signals and nucleotide distribution of *Chlamydomonas* is very different from Arabidopsis and rice [16]. Most significantly, the dominant pattern of NUE signal state of *Chlamydomonas* is pentomer TGTAA in about 52% of genes, while in rice and Arabidopsis is the hexamer AATAAA in only about 10% of genes. It also seems that there is weak FUE (far upstream element) signals with a distinct high G content in *Chlamydomonas* ([16] and Fig.2). The poly(A) site pattern of Arabidopsis and rice is YA (Y=C or T) oriented, while that of *Chlamydomonas* is only one nucleotide A. Therefore, PASS can not be applied directly to *Chlamydomonas*.

In this paper, based on PASS, an updated version of PASS called PASS_VAR is developed, which does not change the original mode of operation, but can be applied to different species through parameter file to set the parameters. PASS_VAR allows user to set his own target states and number states. The length of each state, state type, probability calculation methods can also be customized. Moreover, PASS_VAR Offers more probability calculation methods to calculate the state output probability, such as heterogeneous first-order Markov sub-model and weights of signal patterns.

4 Poly(A) Sites Recognition of *Chlamydomonas*

4.1 Determine the Signal States

Based on the nucleotide distribution and other research results [16], the signal location of *Chlamydomonas* were obtained (Fig. 2).

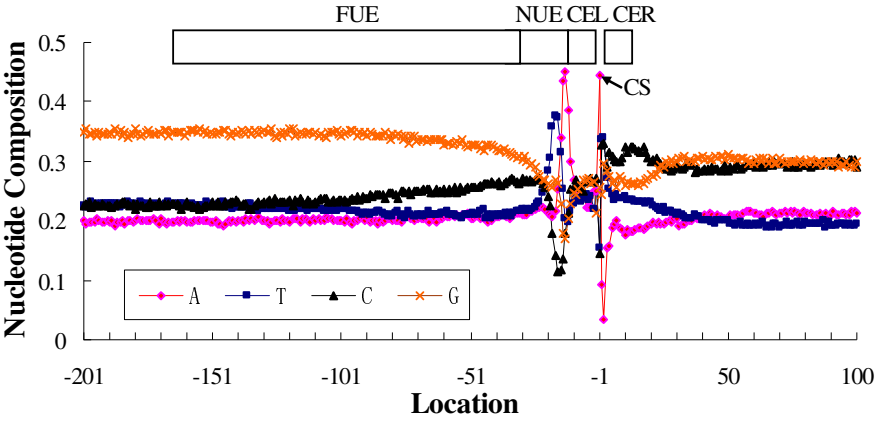


Fig. 2. Nucleotide distribution and signal state location around *Chlamydomonas* poly(A) sites

4.2 Group Sequences by Sequence Patterns

In plant poly(A) signals, NUE signals are the most conservative. TGTAA is especially conservative in the NUE region of *Chlamydomonas*, the proportion of which is up to 52%. Here, 17k sequences were grouped by NUE patterns to set the parameters of each group separately, as shown in Table 2.

Table 2. Five groups of NUE

Group	Dominant 5-grams on NUE	Number of sequences
group1	TGTAA	8622
group2	GTAAC, CTGTA	492
group3	GTGTA,ATGTA,TTGTA,GTAAA,GTAAG,GTAAT	1106
group4	AGTAC,TGCAA,CGTAT,AGTAT,GGTAT	981
group5	None of above	3998

4.3 Parameter Settings of Signal States

Since the signal states FUE and CE are of little conservation, we merged the sequences of group 2 and group 5 defined in Table 2 into a larger group to improve the computational efficiency. Count the top 100 patterns of FUE and CE to obtain their output probability of each base, as shown in Table 3.

Table 3. Signal states of FUE and CE and their output probability of each nucleotide

	Group1				group2 to 5			
Signal	**A**	**T**	**C**	**G**	**A**	**T**	**C**	**G**
FUE	0.08307	0.17949	0.21733	0.52011	0.09063	0.19384	0.20574	0.50979
CEL	0.2556	0.22833	0.25234	0.26374	0.18707	0.20537	0.29689	0.31067
CER	0.06381	0.2169	0.41362	0.30567	0.06039	0.17572	0.45892	0.30497

For the higher conservative signal states NUE and CS, the parameters were calculated from Group 1 to Group 5 separately, and build a heterogeneous first-order Markov model to characterize the characteristics of these two signals. First, a set of signal patterns is required for the sub-model. The CS state is of length 1, so all patterns can be directly used to build sub-models. While the NUE signal patterns are of length 5, the parameters may be mixed with random background noise if total 1024 (4^5) 5-grams are used. In this paper, an entropy [18] based feature extraction method was used to select effective 5-grams.

Given a sequence of length L and a window x(l) of length l, define the duplicate level of x(l) in L as:

$$F_L(x(l)) = T_L(x(l)) / W_L(l) \tag{4}$$

Where $T_L(x(l))$ is the occurrence of x(l) appearing in the L; $W_L(l)$ is the total number of the windows of length l in L. In the following entropy analysis, the sequence patterns were characterized using duplicate level.

The process of selecting 5-grams in NUE state is as follows

1) Initiation: number of samples N1=N2, total number of samples N0=N1+N2, class U=(U1,U2), number of class m=2, number of fragments N.

2) For each 5-gram, repeat the following steps.

2.1) Traverse all samples to obtain duplicate level of each sample f.

a) From upstream 10nt to 30nt of the poly(A) site, slide a scanning window, count number of the occurrences of *f* in all sample sequences as $T_L(f)$.

b) Count the total number of slide windows $W_L(f)$.

c) Calculate duplicate level $T_L(f) / W_L(f)$.

2.2) Count the number of sample f belonging to class Ui and in the range of r(k) as N(k,i), (k = 1,2, ..., N, i = 1,2).

2.3) Count the number f in the range r(k) in all samples as N(k).

2.4) Calculate the probability of sample f belonging to r(k) in class Ui as $p(k,i)$.

2.5) Calculate the probability of a sample belonging to r(k): $p(k)$=N(k)/N0.

2.6) Calculate entropy:

$$H(f) = -\sum_{k=1}^{N} P(k) \sum_{i=1}^{m} P(k,i) \log_2 p(k,i) \tag{5}$$

3) Output the entropy of all 5-grams in ascending order and obtain the 5-grams with entropy less than 1 to calculate the frequency of transfer matrix.

4.4 Parameter Settings of Background States

Within the coverage of each background state, calculate the output probability of each nucleotide of group 1 and the integrated group (group 2 to 5), as shown in Table 4.

Table 4. Background states and their output probability of each nucleotide

	group1				group2 to 5			
BG	**A**	**T**	**C**	**G**	**A**	**T**	**C**	**G**
BG1	0.19639	0.22554	0.23073	0.34734	0.19952	0.22434	0.22523	0.35091
BG2	0.20441	0.21094	0.24821	0.33644	0.20357	0.21533	0.24783	0.33326
BG3	0.26835	0.23135	0.24865	0.25165	0.23635	0.2367	0.26783	0.25912
BG4	0.23335	0.24815	0.25555	0.26295	0.2328	0.2352	0.28285	0.24915
BG5	0.18346	0.24335	0.30862	0.26457	0.17903	0.23765	0.31715	0.26617
BG6	0.19713	0.21242	0.28997	0.30048	0.20126	0.21277	0.29821	0.28777

5 Results

5.1 Performance Indicators

At the nucleotide level, the most common two performance indicators are sensitivity (Sn) and specificity (Sp).

$$Sn=TP/(TP+FN);\ Sp=TN/(TN+FP) \tag{6}$$

Here, TP means true positive, FN is false negative, TN is true negative and FP stands for false positive.

5.2 Sn and Sp

For Sn and Sp calculations, 100 sequences were randomly selected from group 1 to group 5 to calculate Sn, using the corresponding parameters of each group. Another 100 sequences were randomly selected from the control sets for Sp calculation, using parameters of group 5. Moreover, 1000 sequences randomly selected from group 1 to group 5 according to their proportion in the 17k dataset (570,40,70,60,260 sequences,

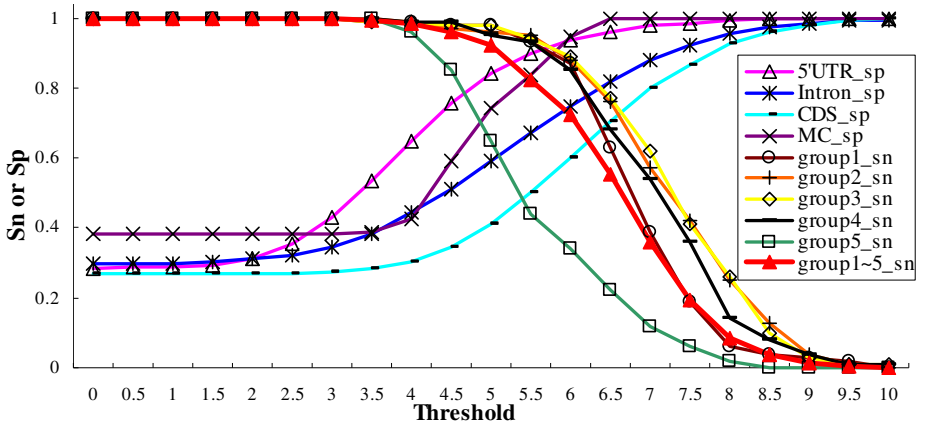


Fig. 3. Sn and Sp of each group. MC: random sequences generated by Markov Chain model. CDS: coding sequences. Group definition is given on Table 2.

respectively) are used to calculate Sn (group1~5_sn). As shown in Fig. 3, Sn and Sp results are similar among the groups containing NUE patterns (group 1 to group 4), while that of the group without NUE pattern (group 5) was significantly lower.

The higher the Sn and Sp is, the better the prediction is. However, the sn and sp can not be increased at the same time, so we define a cross value which is the Y value of the intersect point of Sn and Sp curves to better evaluate our prediction results. The cross values of each group and the integrated group are shown in Table 5, which also indicate that the recognition results of group containing NUE pattern are significantly better than results of non-NUE pattern group. This demonstrates that the NUE pattern plays an important role in poly(A) sites recognition. And although the proportion of the dominant NUE pattern TGTAA is up to 52%, other NUE patterns are also very influential to poly(A) sites identification. Since there is no dominant NUE pattern in group 5, the recognition results of which are relatively poor. However, over 74% of the sequences can be predicted with high accuracy.

Table 5. Cross values of Sn and Sp of each group

Cross value	group1	group2	group3	group4	group5	group1~5
5UTR	0.93	0.91	0.93	0.91	0.84	0.87
MC	0.91	0.9	0.9	0.89	0.78	0.83
Intron	0.88	0.8	0.8	0.78	0.72	0.74
CDS	0.85	0.72	0.72	0.7	0.63	0.65

5.3 Average Score

By using PASS_VAR, for each nucleotide sequence, each location corresponds to a score indicating the possibility being a poly(A) site. The average score of the sequences of each group is calculated, as shown in Fig. 4, the control sequences are more evenly that there is no particularly prominent score, which is consistent with the fact that the control sequence does not contain poly(A) sites. In contrast, the scores of

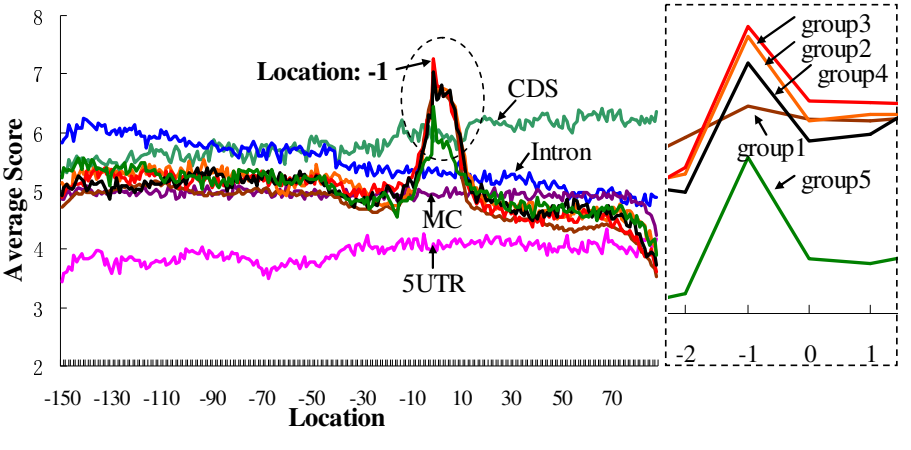


Fig. 4. Average score of each group

the poly(A) sites (location=-1) are all significantly higher than the scores of other locations, even if in the group 5 which has the worst identification result. This result also indicates the effectiveness of our identification model.

6 Conclusions

In this paper, poly(A) sites of *Chlamydomonas* were identified using GHMM-based recognition model combining the entropy based feature selection method. The experimental results show the effectiveness of the model as well as the parameter statistics method of grouping sequences by NUE patterns. Most of the poly(A) sites can be identified using the present method, the most important ones of which can be verified through biological experiments to reduce workload, thus our model is of high practical value in biological experiments and genomic analysis.

In addition, due to that the plant and alga poly(A) signals are weak and of large variation, the positioning of poly(A) sites is a very difficult task. Besides, the poly(A) signals and nucleotide distribution of *Chlamydomonas* are very different from those of Arabidopsis and rice. Our improved poly(A) sites identification system is of strong operational and broad applicability, which has been applied on *Chlamydomonas* in this paper, Arabidopsis [15] and rice [17] and allows adjusting the parameters according to the characteristics of the surrounding sequences of target species to adapt to their own poly(A) sites identification.

Acknowledgments

This project was funded by grants from the National Natural Science Foundation of China (No. 60774033), Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 20070384003 and 20090121110022), and Xiamen University's National 211 Project 3rd period (No. 0630-E62000).

References

1. Proudfoot, N.: New perspectives on connecting messenger RNA 3' end formation to transcription. Curr. Opin. Cell. Biol. 16, 272–278 (2004)
2. Edwalds-Gilbert, G., Veraldi, K.L., Milcarek, C.: Alternative poly(A) site selection in complex transcription units: Means to an end? Nucleic Acids Res. 25, 2547–2561 (1997)
3. Zarudnaya, M.I., Hovorun, D.M.: Hypothetical double-helical poly(A) formation in a cell and its possible biological significance. IUBMB Life 48, 581–584 (1999)
4. Kan, Z.Y., Rouchka, E.C., Gish, W.R., States, D.J.: Gene structure prediction and alternative splicing analysis using genomically aligned ESTs. Genome Res. 11, 889–900 (2001)
5. Legendre, M., Gautheret, D.: Sequence determinants in human polyadenylation site selection. BMC Genomics 4 (2003)
6. Tabaska, J.E., Zhang, M.Q.: Detection of polyadenylation signals in human DNA sequences. Gene 231, 77–86 (1999)
7. Ji, G.J., Wu, X.H., Zheng, J.T., Shen, Y.J., Li, Q.Q.: Modeling plant mRNA poly(A) sites: Software design and implementation. Journal of Computational and Theoretical Nanoscience 4, 1365–1368 (2007)

8. Ji, G.J., Wu, X.H., Jiang, Y.H., Li, Q.Q.: Implementation of a Classification-Based Prediction Model for Plant mRNA Poly(A) Sites. Journal of Computational and Theoretical Nanoscience 7, 1–6 (2010)
9. Hu, J., Lutz, C.S., Wilusz, J., Tian, B.: Bioinformatic identification of candidate cis-regulatory elements involved in human mRNA polyadenylation. RNA Society 11, 1485–1493 (2005)
10. Hajarnavis, A., Korf, I., Durbin, R.: A probabilistic model of 3' end formation in Caenorhabditis elegans. Nucleic Acids Res. 32, 3392–3399 (2004)
11. Loke, J.C., Stahlberg, E.A., Strenski, D.G., Haas, B.J., Wood, P.C., Li, Q.Q.: Compilation of mRNA polyadenylation signals in Arabidopsis revealed a new signal element and potential secondary structures. Plant Physiol. 138, 1457–1468 (2005)
12. Mayfield, S.P., Manuell, A.L., Chen, S., Wu, J., Tran, M., Siefker, D., Muto, M., Marin-Navarro, J.: Chlamydomonas reinhardtii chloroplasts as protein factories. Curr. Opin. Biotechnol. 18, 126–133 (2007)
13. Wilson, N.F., Iyer, J.K., Buchheim, J.A., Meek, W.: Regulation of flagellar length in Chlamydomonas. Semin. Cell. Dev. Biol. 19, 494–501 (2008)
14. Rupprecht, J.: From systems biology to fuel–Chlamydomonas reinhardtii as a model for a systems biology approach to improve biohydrogen production. J. Biotechnol. 142, 10–20 (2009)
15. Ji, G.L., Zheng, J.T., Shen, Y.J., Wu, X.H., Jiang, R.H., Lin, Y., Loke, J.C., Kimberly, M.D., Reese, G.J., Li, Q.Q.: Predictive modeling of plant messenger RNA polyadenylation sites. BMC Bioinformatics 8 (2007)
16. Shen, Y.J., Liu, Y.S., Liu, L., Liang, C., Li, Q.Q.: Unique features of nuclear mRNA Poly(A) signals and alternative polvadenylation in Chlamydomonas reinhardtii. Genetics 179, 167–176 (2008)
17. Shen, Y.J., Ji, G., Brian, J.H., Wu, X.H., Zheng, J.T., Greg, J.R., Li, Q.Q.: Poly(A) signals and extensive alternative polyadenylation revealed through genome level analysis in rice. Nucleic Acids Res. 36, 3150–3161 (2007)
18. Lin, S.K.: Diversity and Entropy. Entropy 1, 1–3 (1999)

A Study to Neuron Ensemble of Cognitive Cortex ISI Coding Represent Stimulus

Hu Yi and Xin Tian*

Research Center of Basic Medicine, Tianjin Medical University, Tianjin 30070, China
tianx@tijmu.edu.cn

Abstract. According to the Hebb's cell assemble theory about memory, multi-neurons, encoding and expressing information (stimulus) by coorperating firing, assemble to the functional neuronal ensemble. Traditional neurons coding is the average frequency coding, but this method loss most dynamic information because the window is too large. The purpose of this thesis is to raise and perform ISI coding on the small time scale, to make up the limitation of the average frequency coding. The results may help to support the study of neuronal computing for the neuronal ensemble coding.

Keywords: Neuronal ensemble, Spatiotemporal sequences, Neuronal population, ISI coding, Average frequency coding, Stimulus.

1 Introduction

To realize the nervous system function, the brain is an effective information processing and transmission. Neurons in the brain is the basic functional units of a large number of clusters consisting of neurons in complex neural networks involved in sensation, cognition and memory and other advanced features of the brain. Research information between neurons is how to pass and the code is hot Neuroscience.

Neurons in electrical signals and chemical signals used for transmission of information between cells, neurons with a cross-membrane voltage difference, that is, resting potential. When neurons were normal resting above threshold depolarizing stimulus, the transmembrane voltage difference into a rapid action potential, it can, through an axon from the cell site to another part of the expansion, and rapid to make long-distance transmission, occurred in the neurons of the action potential, its amplitude and time is fixed. Neuronal action potential is the primary means of transmission of information, one is usually between neurons through a number of action potential discharge sequences composed of information and communication.

From the informatics point of view, the neuron per unit time or the frequency of issuance of the release of the time contains a wealth of information, how to analyze and code the information room in recent years the development of neural science and information science one of the hot crossover study.

The average frequency coding is the classic method of neuron ensemble coding, which is defined as a certain period of time in neurons the average discharge frequency

* Corresponding Author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 27–32, 2010.

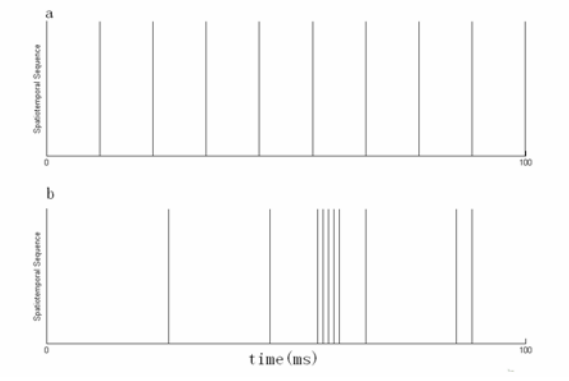


Fig. 1. The two groups with different distri-bution of the same frequence. Two groups of neurons in the release of 100ms window are 10 times, but the distribution of different: (a) in the 100ms window for the uniform payment, (b) within the 100ms window for the nonlinear discharge. The average frequency of two groups the results were consistent coding, all of 10, but the distribution is clearly different, in other words, the average frequency of encoding information to cover up the details.

strings neurons determines the information contained in string. The advantage is simple, fast, intuitive.

As a result of a longer window length, neurons release the details of information can be lost. As the examples, the same window (100ms) were paid 10 times the number, but their distribution in the window is significantly different, Fig.1 (a) for the uniform distribution of window, (b) concentrated in the middle of the frequency coding used by the statistics of each group the number of the release window is the same, are 10 times, that is to say, all neurons in the details of the level of release of information is covered.

To make up for less than the average frequency of coding, we consider another approach to the characterization of coding neurons release the details of information, as well as clusters of neurons among the neurons relevant information, which is ISI (InterSpikeInterval, neurons string action Fengfeng potential interval) encoding.

2 Methods

Abeles developed the Time Code in 1982, the main difference between the time coding and the frequency coding is the time coding consider the relevance of time.Time coding theory suggests that neurons of the time series model contains a wealth of dynamic information. One important type of time coding is the ISI coding, which is based on neuron action potentials in spike trains of the variation interval Fengfeng basic information for the coding. We focus on the details of neuronal release of information, so here we select the small window-related methods. Concrete steps are as follows.

(1) To calculate the ISIs of the spatiotemporal sequences of neuronal firing.
(2) Neurons in the value of ISIs Ordinate, abscissa point in time for neurons to map out the dispersion diagram of the ISI.

(3) Fitting for each neurons' ISI values to the curve.
(4) On the fitted curve to 1ms sampling frequency for re-sampling.
(5) The highest frequency as the reference group, other groups for the window width to 20ms, 20ms retrieve the data, that is, 20 points, and reference data for the related group, to be related to values, and take the absolute value of the related.
(6)10ms for the step to move the window, repeat the step until all the data statistics.
(7)Normalize all related value and then to plot the dynamics topographic map.

3 Results

The following are sixteen of the ISIs before and after the effective stimulus coded topographic map (Fig.2). The abscissa is time, and the red triangle is the points of stimulation. Longitudinal coordinates for the number of neurons, a total of 120 neurons. The average associated value before and after the stimulating is 0.21±0.06, and the average associated value during the stimulating is 0.65±0.07.

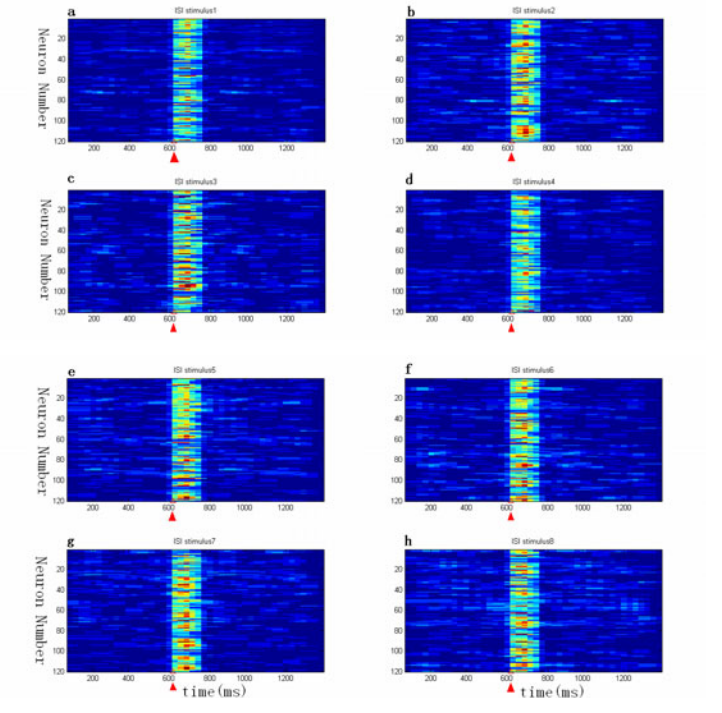


Fig. 2. ISI coding dynamic topographic maps. The red triangle is the stimulated moment, transverse axis is time (ms), vertical axis for the number of neurons. (a) to (p) were the first to the sixteen of the ISI coding stimulate dynamic topographic maps.

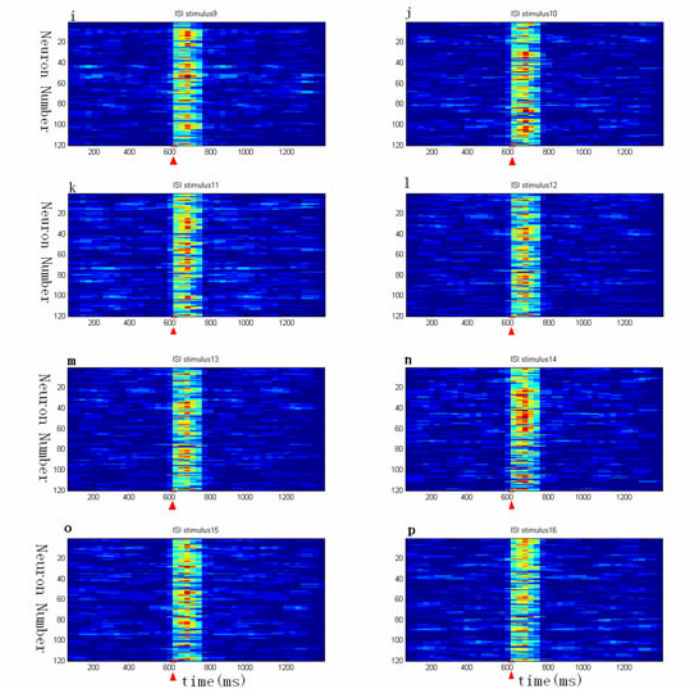


Fig. 2. (*Continuted*)

From the sixteen-coded ISI topographic maps, we can see the color turned from blue to red during the stimulating, this means these neurons in the discharge frequency increased after stimulation that excitatory neurons increased the level of activity to increase, that is, these neurons in the stimulated cells formed ensemble, a common response to this stimulus.

4 Discussion

4.1 The Advantages and Disadvantages of Both Methods

The average frequency of the advantages of coding is simple, intuitive, easy to implement, the disadvantage of a larger window width can not compare the level of information detail. ISI coding is able to distinguish between the merits of the information level of detail, and can reflect the cell clusters of neurons within the various interrelationships between them.

4.2 ISI Coding Order Curve Fitting of the Selected

Order polynomial fitting the needs of the neurons according to data flexibly select different circumstances, when used in the actual end of our constituency to the dozens of bands from several bands ranging from, if necessary, can be taken in sub-fitting

method to achieve in order to ensure the best fitting results. Would also like to note that if the neurons in a small amount of data, for example, only 2-3 data points, which needs to be rounded down to the nearest, it is because if too few points, with its fitted curve is wrong meaningless.

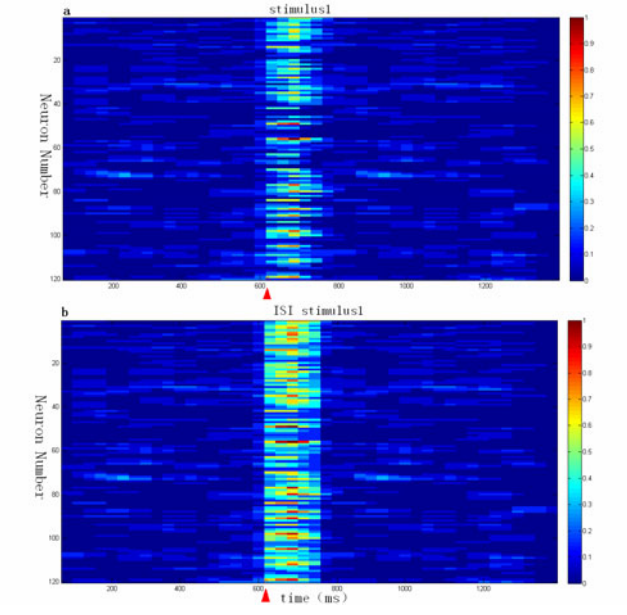


Fig. 3. The comparison of average frequency coding and ISI coding.(a) is the average frequency of the dynamic topographic map encoding, (b) is encoded ISI dynamic topographic maps. ISI coding can be seen change more evident than the average frequency coding, it can better reflect the neurons to stimulate the process of issuing the details of the dynamic characteristics of the changes.

5 Conclusion

This paper studied the neuron ensemble ISI and average frequency coding ,and comparison of the effective stimulation of electrical activity of neurons encoding cluster studies, this paper's main findings are as follows:

(1) The ISI coding method of neuronal ensemble can effectively encode the acticities of the neuronal under the stimulation, and have good robustness:

The average associated value before and after the stimulating is 0.21±0.06,and the average associated value during the stimulating is 0.65±0.07.

The ISI dynamic topography mapping (Fig.2) can be intuitive to see: the color turned from blue to red during the stimulating.This means these neurons in the discharge frequency increased after stimulation that excitatory neurons increased the level of activity to increase, that is, these neurons in the stimulated cells formed ensemble, a common response to this stimulus. The 16 results showed that stimulation of neurons encoding the same or less, indicating that the method is robust.

(2) The ISI coding method of neuronal ensemble can be used to express instaneous patterns of neuronal ensemble dynamic encoding on the small time scale(<20ms) to make up the limitation of the average frequency coding.

The average frequency of coding is a classic, easy way, but because its window is too large, the majority of the loss of dynamic information, is not an effective characterization of electrical activity of neurons in the details of cluster information. In this paper, the ISI and the encoding time due to the small-scale (<20ms), they are able to effectively compensate for the average frequency of coding this shortcoming.ISI code will be drawn by the dynamic topographic map and the average frequency of the dynamic code map topographic map (Fig.3) can be seen compared, ISI encoded resolution than the average high-frequency coding, the process can better reflect the stimulation neurons release the details of the dynamic characteristics of the changes.

Acknowledgement

This work was supported by the National Natural Science Foundation of China. (30770545).

References

1. Wang, J.-F., Zhou, Y.: A Novel Method for Multi-channel Neuronal Spike Detection and Classification. Progress in Riochemistry and Riophysics, 641–647 (2009)
2. Zheng, X.-Y.: Clusters of Neurons Encoding the Spatial and Temporal Study of The Status. Miscellaneous International Biomedical Engineering, 186–188 (2007)
3. Chen, N., Wu, Y., Wang, J.-H.: Correlation Action Between The Refractory Periods and Threshold Potentials and The Spike Programming In Cortical Neurons. Chin. J. Appl. physiol. 24 (2008)
4. Wang, R.-B., Yu, W.: To Stimulate The Neurons Under The Group of Large-scale Stochastic Nonlinear Evolution Model and The Dynamic Neural Series. Journal of Biomedical Engineering, 243–247 (2006)
5. Wang, J.-Y., Luo, F., Zhang, H.-Y.: Awake Rat Anterior Cingulate Cortex Neurons in The Injury Response Group. Journal of Beijing University, 47–52 (2004)
6. Halbach, M., Egert, U., Hescheler, J.: Estimation of Action Potential Changes From Field Potential Recordings in Multicellular Mouse Cardiac Myocyte Cultures. Cell. Physiol. Biochem., 271–284 (2003)
7. Wang, X.-Q.: Neural Coding Strtegies in Auditory Cortex. Hear Res., 81–93 (2007)

STDP within NDS Neurons

Mario Antoine Aoun

Beirut, Lebanon
maaoun@ndu.edu.lb, marioaoun@hotmail.com

Abstract. We investigate the use of Spike Time Dependent Plasticity (STDP) in a network of Nonlinear Dynamic State (NDS) Neurons. We find out that NDS Neurons can implement a form of STDP; a biological phenomenon that neocortical neurons own, and would preserve their temporal asymmetric windows of firing activity, while stabilizing to Unstable Periodic Orbits, called UPOs, considered as their neural states. Such correlation and ease of integration by using STDP within NDS neurons show that those NDS neurons can truly implement biological realism through their dynamics as it was early speculated in their invention in 2005.

Keywords: NDS neuron, Chaotic neuron, Chaos control, UPOs, Chaotic neural networks, Chaotic spiking neural networks, Chaotic spiking neurons, STDP.

1 Introduction

Since a stabilized UPO can be depicted to a memory address domain as in neuromorphic engineering [1], or can take other important relevance due to its theoretical evidence [2], and since "Chaos may be the chief property that makes the brain different from an artificial-intelligent machine" as hypothesized by Freeman [3], then it is envisaged that UPOs; presented by windows of temporal spike patterns, can be designated as neural states and communicated between chaotic neurons called Nonlinear Dynamic State – NDS – Neurons [4]. We have to note that NDS neuron self stabilization property inherits the latest advancements of Delay Feedback Control – DFC – theory which was invented by Pyragas in 2003 [2]; a breaking through theory in chaos control.

NDS neurons are able to stabilize their internal dynamics using feedback control through connection time delays [4], this occurs by implementing unstable degrees of freedom with bounded parameters on their feedback loops [2]. UPOs considered as neural states and presented by periodic spike patterns (sequences of binary bits of 0s and 1s), could be used to enhance neural information processing within artificial neurons [4], this is since their dynamical reservoir is mathematically proven to be infinite [2].

In section 2, we review the NDS Neuron model and we present a Hopfield like network architecture, which we'll study; consisting of three NDS Neurons recurrently connected. In Section 3, the STDP learning rule is described and implemented within the network. The purpose of this implementation is to make the neurons synchronize

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 33–43, 2010.

and stabilize their internal dynamics in accordance to their synaptic input. Experimental results and time series are presented in section 4. In the last section (Section 5), we discuss further evidence of such modeling; of NDS neurons neural states, to be interpreted as "memory binding". Memory binding is achieved through temporal coding and synaptic plasticity, taking advantage of the rich chaotic reservoir of NDS neuronal states and their synchronization.

2 The NDS Neuron Model and Network Architecture

The Nonlinear Dynamic State Neuron - NDS Neuron – was invented by Nigel Crook et al. and presented in the European Synposium on Artificial Neural Networks in April 2005 [4]. The NDS Neuron model is different than classical Neuron models like threshold gates and perceptrons which have binary states and finite memory capacity; instead its memory capacity consists of the number of Unstable Periodic Orbits – UPOs – that can be stabilized within its internal dynamics. A Referential memory or - generally speaking - a dynamic state of the NDS neuron corresponds to a stabilized UPO, consisting of "a temporal periodic output pattern of spikes" [5]. This means, and as hypothesized in [5], that "the NDS neuron has at its disposal a very rich range of internal states together with a *vocabulary* for communicating these states to other neurons" [5]. With slight modification in the interpretation of the NDS neuron, we'll review the NDS neuron [4] system of dynamical equations next.

The dynamical state of the NDS neuron at time *t* is governed by three ODEs, described as it follows:

$$u_i(t) = u_i(t_{pre}) + I_i(t)$$

$$x_i(t) = x_i(t-1) + b(-y_i(t-1) - u_i(t-1)) \tag{1}$$

$$y_i(t) = y_i(t-1) + c(x_i(t-1) + a y_i(t-1))$$

With initial conditions:

$$x_i(0) = 0$$
$$y_i(0) = 0$$
$$u_i(0) = n_0$$

Where,

n_0 is a random number between -1 and 0.

$$u_i(t_{pre}) = u_i(t-1) + d(v + u_i(t-1)(-x_i(t-1)) + k u_i(t-1)) \tag{2}$$

a, b, c, d, v and *k* are constant parameters *(a=0.002, b=0.03, c=0.03, d=0.8, v=0.002, k=-0.057)*

$u_i(t)$ is the action potential of the NDS Neuron *i* at time *t*,

$u_i(t_{pre})$ is the internal voltage during potentiating process (t_{pre}),

$I_i(t)$ is the total potential input; that the neuron is receiving, integrating and conducting at its synapse - is maintaining at its synapse at accordant points in time as:

$$I_i(t) = \sum_{j=1}^{n} w_{ij}(t)\gamma_j(t-\tau_{ij}) \tag{3}$$

Where, n is the number of connections that the *ith* neuron has, $w_{ij}(t)$ denotes the weight from neuron j to neuron i, τ_{ij} is the connection time delay (which works as feedback control) from neuron j to neuron i and $\gamma_j(t-\tau_{ij})$ being the spike output of neuron j at time step $t-\tau_{ij}$.

$\forall r \in (N = \textit{Number of Neurons in the network})$,

$$\gamma_r(t) = \begin{cases} 1, u_r(t) > \theta \\ 0, u_r(t) \le \theta \end{cases} \tag{4}$$

θ is the threshold and equal to zero.

When the NDS neuron i fires (i.e: when $u_i(t) > \theta$) then $u_i(t) = n_0$ where n_0 is called the after-spike reset value of $u_i(t)$.

The state of an NDS Neuron, when it is in isolation, is chaotic (Fig. 1).

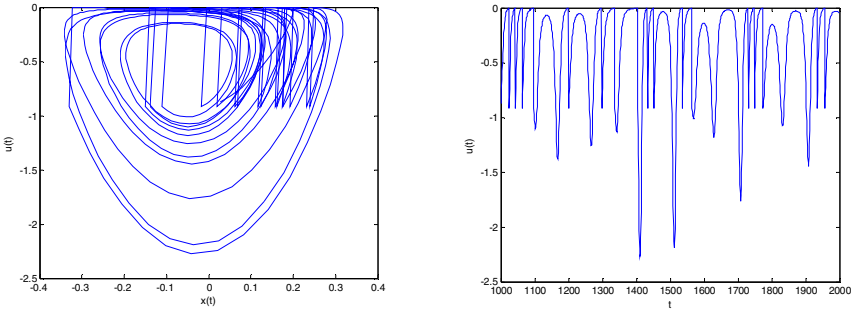


Fig. 1. Phase space plot (a) and time series data (b)

We present a network of three NDS Neurons recurrently connected (Fig. 2). All connection delays between NDS Neurons are equally set to constant time delay different than zero ($\tau_{i,j} <> 0 \; \forall i, j \in N$; N = 3 which is the number of neurons in the network), except for $\tau_{0,0}$, $\tau_{1,1}$, $\tau_{2,2}$ which are not present and set to zero (i.e: In this architecture, NDS neurons only have recurrent connections between themselves and don't have self feedback connections).

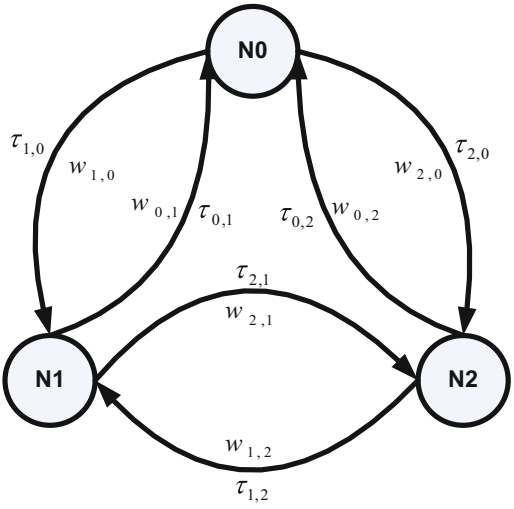


Fig. 2. Network Architecture

3 Synaptic Equations

Recent studies in neuroscience [6] suggest that the exact firing time between neurons has much more influence on information processing rather than the firing rates of these neurons; as it was previously thought ([7], [8]). Additionally, the concurrency of firing times that occurs between neurons is a major feature for information binding, memory retrieval and temporal coding inside the brain [9]. If multiple neurons fire synchronously then their pulses or spikes arrive at a target neuron with the same firing time, thus causing the last to fire with a greater probability than being pulsed with multiple spikes randomly or at different times [9].

In this research and for our concern in the construction of the synaptic learning process that we implement at the synapse between NDS Neurons, we will consider Spike-Timing-Dependent Plasticity (STDP) learning rule [10], a recent biological discovery in Hebbian Learning theory and Synaptic Plasticity which is based on the time of the firing of a neuron and not on the rate of the firing of a neuron as it was previously considered. We mention this rule in its simplest form, which can be stated as:

"synapses that are activated slightly before the cell fires are strengthened whereas those that are activated slightly after are weakened" [11]

In biological neural information processing, this can be analyzed as the pulse; temporally embedded in the information being processed on the target neuron, is received instantly when or before this target neuron fires i.e: at expected times of firings of this neuron (this may be considered like the pulse is available in the reservoir of temporal spikes' window which constitutes the response properties of this neuron). This pulse is considered to confirm that the information being processed at this instant in time is

relevant to the adequate firing patterns of the target neuron and the last should take an action by changing its state to another state having more correlations with the presented input.

In the model of synaptic learning presented herewith, we removed inhibitory connections (negative weights), considering the fact that the learning rule "should strengths causal interactions in the network" [9]. Furthermore, high frequency pre-synaptic activity could lead to long term potentiation (LTP) at the synapse; a major direction in our implementation and this is a fundamental mechanism behind neurons' assembly inside the hippocampus of the mammalian brain, forming what is so called cerebral plasticity and commonly known as long term memory. Also, it was proven in [12] that if high and repetitive pre-excitation; from pre synaptic neurons in the hippocampus and cortical neurons, called excitatory post synaptic potentials (EPSPs), occur slightly before a neuron fires, they cause an increase in the synaptic activity and an increase in the action potential of the neuron, forming LTP [12]. But if EPSPs occur after the action potential then it forms Long Term Depression (LTD).

Accordingly, this can be interpreted and implemented, in our model, like the following,

At time t=0, the weight which presents the synapse strength is:

$$w_{ij}(0) = \textit{Random number between 0.05 and 0.3}$$

At t>0, and when an impulse is received to the target neuron, the target neuron will change the weight of the synapse according to the following:

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t_{pre}) \text{ , where } \Delta w_{ij}(t_{pre}) \geq 0 \tag{5}$$

We have to note that w_{ij} is never weakened. This doesn't mean that the NDS neuron won't have a form of long term depression because the dynamical equation of u(t), considered as internal voltage of the neuron, is not affected by pre-synaptic input of voltage spikes only, but with other dynamical variables like x(t), which works on regulating (i.e: also inhibiting and controlling) the action potential of the NDS Neuron.

We consider a leaky integrator at the synapse elicited with the following synaptic plasticity equation:

$$\Delta w_{ij}(t_{pre}) = (-u_i(t_{pre}) - I_i(t_{pre}))P(u_i(t_{pre}), I_i(t_{pre}))\gamma_{ij}(t-\tau_{ij}) \tag{6}$$

Where,

$-u_i(t_{pre})$, being the membrane voltage, taken as the inverse of the internal voltage $u_i(t_{pre})$.

$I_i(t_{pre})$, being the total input voltage from other neurons, presented through the synapses, showing pre-synaptic activity while occurring during pre synaptic potential phase (i.e: in the potentiating process; while the neuron is firing or very near to its peak action potential to fire):

$$I_i(t_{pre}) = \sum_{j=1}^{n} w_{ij}(t-1)\gamma_j(t-\tau_{ij}) \tag{7}$$

$P(u_i(t_{pre}), I_i(t_{pre}))$, being interpreted as activation/desactivation variable in the simplest form of a STDP learning rule, excluding weight decay or any exclusive inhibition effect from the input:

If $u_i(t_{pre}) < \theta$ and $u_i(t_{pre}) + I_i(t_{pre}) > \theta$ then P = 0 L.1

If $u_i(t_{pre}) < \theta$ and $u_i(t_{pre}) + I_i(t_{pre}) < \theta$ then P = 1 L.2

If $u_i(t_{pre}) > \theta$ and $u_i(t_{pre}) + I_i(t_{pre}) > \theta$ then P = -1 L.3

If $u_i(t_{pre}) > \theta$ and $u_i(t_{pre}) + I_i(t_{pre}) < \theta$ then P = 0 L.4

Note that the threshold is equal to zero $(\theta = 0)$, the learning rules (L.1, L.2, L.3 and L.4) described above would imply:

L.1 implies: Spike is not expected; the neuron may fire but won't update its synapse.

L.2 implies: Spike is received in phase time, (expected); the neuron won't fire but updates its synapse.

L.3 implies: Spike is received at exact time (as expected); the neuron will fire and updates its synapse.

L.4 implies: Inhibitory Spike (does not occur); the neuron won't fire, and won't update its synapse.

$\gamma_j(t - \tau_{ij})$, which is the spike output of neuron j at time $t - \tau_{ij}$, is interpreted as the maximum conductance of this firing channel, acting at a specific moment in the temporal spikes window τ_{ij} of neuron j.

Note that $I_i(t_{pre})$ is being catalyzed due to the presence of an impulsive current *and have to remain in equilibrium with internal potential changes and should maintain conductance equilibrium at the synapse at transduced specific firing moments available in the post synaptic neurons' set of temporal windows (UPOs)*, thus and after synaptic changes (i.e: weight updates) , the following reaction takes place:

$$I_i(t) = \sum_{j=1}^{n} ((w_{ij}(t-1) + \Delta w_{ij}(t_{pre}))\gamma_j(t - \tau_{ij})) \quad (8)$$

Since,

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t_{pre}) \quad (9)$$

Then and finally, the result of all synaptic changes; that affects the internal voltage of the neuron, after being catalyzed at pre-synaptic activity, would be:

$$I_i(t) = \sum_{j=1}^{n} ((w_{ij}(t))\gamma_j(t - \tau_{ij})) \quad (10)$$

Here above, we have implemented the variables of the synaptic equation which governs the connections dynamics between NDS neurons, while preserving the rich dynamical reservoir of states of the NDS neuron.

4 Experimental Results

We consider a network of three NDS Neurons recurrently connected (Fig. 2). The goal of each neuron is to synchronize temporal patterns with other neurons and to increase its sensitivity to *synchronizing neurons*. Such sensitivity and plasticity is increased through synaptic weights which are intensified at exact and expected spikes times and governed by STDP. The network will run 10000 time steps (t = 0 to t = 10000). It was experimented that 10000 time steps are more than enough to ensure synchronization and stabilization of NDS Neurons states. In this example, the DFC - $\tau_{i,j}$ - used is equal to 100 for all active connection delays.

Initially, the after spike reset value (n_0) of every neuron is set to a random number between -1 and 0:

$$n_0 \text{ of } Ni = \text{Random number between -1 and 0}$$

This means:

$$n_0 \text{ of } N0 = \text{Random number between -1 and 0}$$

$$n_0 \text{ of } N1 = \text{Random number between -1 and 0}$$

$$n_0 \text{ of } N2 = \text{Random number between -1 and 0}$$

At t =0 all NDS neurons will start with:

$$x_i(0) = 0$$
$$y_i(0) = 0$$
$$u_i(0) = n_0$$

With random connection weights:

$$w_{ij} = \text{Random number between 0.05 and 0.3}$$

With Delay Feedback Control = 100

$$\tau_{i,j} = 100 \ \forall i, j \in N \text{ , } i<>j$$

$$\tau_{i,j} = 0 \ \forall i, j \in N \text{ , } i=j$$

In this architecture (Fig. 2), $w_{0,0}$, $w_{1,1}$ and $w_{2,2}$ are not considered because their connections' time delay are removed and set to zero. NDS neurons can have self re-curring connections, but to emphasize stabilization between NDS Neurons by using STDP, self recurring connection weights, and time delays, are not considered in this architecture.

The network will run for 2000 time steps. In this phase (0 <= t <= 2000), NDS Neurons are left in isolation (i.e: not connected) and their output is purely chaotic (Fig. 3).

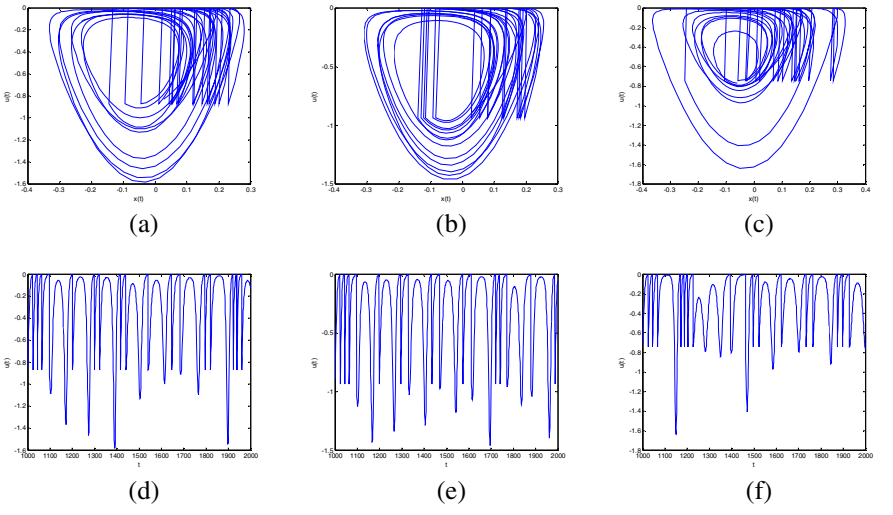


Fig. 3. Phase plots (a), (b), (c) of Neurons N0, N1, N2 in isolation and their time series (d), (e), (f) respectively, showing the chaotic behavior of NDS Neurons when STDP is Off

At t = 2000, connections between NDS neurons are turned on, it was experimented that 3000 thousands steps are enough to insure stabilization. The synaptic weights for nonzero connection delays are updated using STDP at run time, other weights (self recurring) are neither used nor updated – not considered (i.e: set to zero). Synaptic changes will drive NDS neurons to stabilize their internal dynamics into UPOs (Fig. 4).

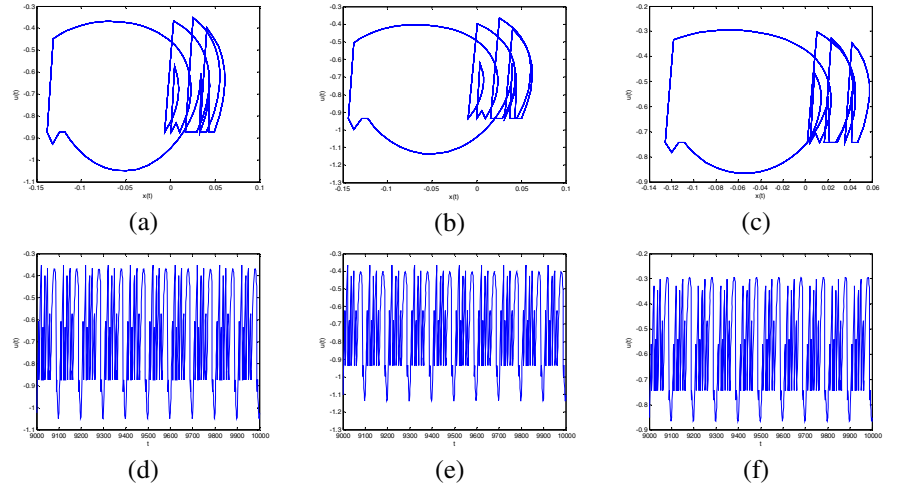


Fig. 4. UPOs (a), (b), (c) of Neurons N0, N1, N2 and their time series (d), (e), (f) respectively, showing synchronizing and periodic output when STDP is used

Synaptic weights of every connection are governed by STDP and are shown in Fig. 5.

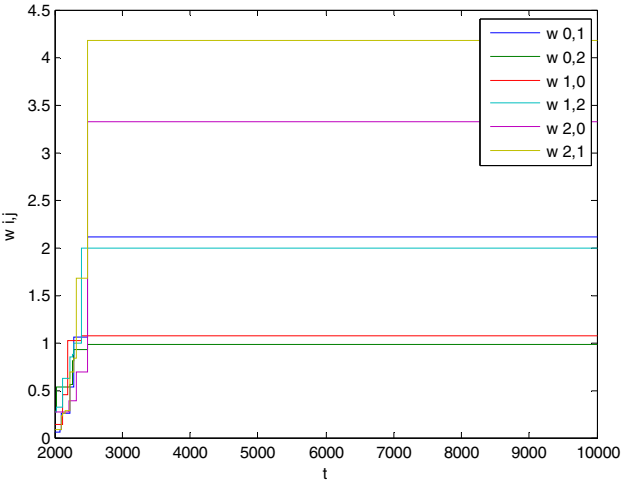


Fig. 5. Weight updates when STDP is turned ON (i.e: starting at time step t = 2000) showing very fast stabilization

5 Discussion

We implemented STDP learning rule in a recurrent network of NDS Neurons. We showed that when STDP governs the synaptic weights of these neurons, then NDS Neurons are able to synchronize and stabilize their internal dynamics to periodic spike output patterns.

We note that the range of the number of UPOs in the chaotic reservoir of a NDS Neuron was experimentally proven to be extremely large [5] and theoretically proven to be infinite [2]. Our analysis, regarding the interesting phenomenon of NDS Neuron states stabilization using STDP, is adequate to Crook speculation [5] of NDS Neurons states communication and vocabulary, as also very relevant in accordance to Izhikevich hypothesis [13] which refers to [14] and states: "Neurons or cortical columns need rhythmic activity to communicate selectively." Also and in [13], we quote: "the frequency of a periodically spiking neuron does not carry any information other than identifying a channel of communication… Information (i.e: neural code) carried through modulations of inter spike intervals." As we consider, very meaningful in analyzing the behavior of NDS neurons state assembly and the communication channels they build using STDP.

We support the network model and the STDP learning rule that we implemented to be based on the laws on cognitive computations by Izhikevich in [9]. On Synchronization, "it should be so rare and difficult to occur by chance that when it happens, even transiently in a small subset of the network, it would signify something important, something meaningful, e.g., a stimulus is recognized, two or more features are bound, attention is paid" [9].

Since we have insured stabilization of NDS Neuron states, our next goal is to extend this research to get advantage of the enormous number of states these neurons can settle onto, and to use it for pattern mapping (e.g: memory binding) and recall. This is based on our vision that NDS Neurons behavior have significance in biological neural information processing [5], and while using STDP, NDS Neurons can embed polychronous groups [9]. This would be considered for future research. Additionally, we quote from [15]: "As to the binding mechanism based on temporal signal correlations, its great advantage - being undemanding in terms of structural requirements and consequently ubiquitously available and extremely flexible - is offset by its quantitative limitations due to the limited bandwidth of neural signals." But by implementing UPOs as neural states, then this limitation is bypassed, since and when NDS Neurons are communicating UPOs via their synapses with STDP and (if also) they are programmed to polychronize [9], then the bandwidth of neural signals becomes unlimited, thus the binding mechanism based on temporal signal correlations would implement "memory binding" within the chaotic spiking neural network of NDS Neurons. This will be the aim, to attain and expand in the next step.

References

1. Deiss, S.R., Douglas, R.J., Whatley, A.M.: A Pulse-coded Communications Infrastructure for Neuromorphic Systems. In: Maass, W., Bishop, C.M. (eds.) Pulsed Neural Networks, pp. 157–178. MIT Press, Cambridge (1999)
2. Pyragas, K.: Time-delayed Feedback Control Method and Unstable Controllers. In: PHYCON 2003 Proceedings of the 2003 International Conference on Physics and Control, vol. 2, pp. 456–467. IEEE Computer Society, Washington (2003)
3. Freeman, W.J.: The Physiology of Perception. Scientific American 264(2), 78–85 (1991)
4. Crook, N.T., Goh, W.J., Hawarat, M.: The Nonlinear Dynamic State Neuron. In: Verleysen, M. (ed.) ESANN 2005 Proceedings of the 13th European Symposium on Artificial Neural Networks, pp. 37–42. D-side Publishing, Belgium (2005)
5. Crook, N.T., Goh, W.J., Hawarat, M.: Pattern Recall in Networks of Chaotic Neurons. BioSystems 87, 267–274 (2007)
6. Beggs, J.M., Plenz, D.: Neuronal Avalanches are Diverse and Precise Activity Patterns that are Stable for many Hours in Cortical Slice Cultures. Journal of Neuroscience 24, 5216–5229 (2004)
7. Rieke, F., Warland, D., de Ruyter van Steveninck, R., Bialek, W.: Spike: Exploring the Neural Code. MIT Computational Neuroscience Series. MIT Press, Cambridge (1997)
8. Shadlen, M.N., Newsome, W.T.: The Variable Discharge of Cortical Neurons: Implications for Connectivity, Computation and Information Coding. Journal of Neuroscience 18, 3870–3896 (1998)
9. Izhikevich, E.M.: Polychronization: Computation with Spikes. Neural Computation 18, 245–282 (2006)
10. Song, S., Miller, K.D., Abbott, L.F.: Competitive Hebbian Learning through Spike-timing-dependent Synaptic Plasticity. Nature Neuroscience 3, 919–926 (2000)
11. Rao, R.P., Sejnowski, T.J.: Spike-timing-dependent Hebbian Plasticity as Temporal Difference Learning. Neural Computation 13(10), 2221–2237 (2001)

12. Paulsen, O., Sejnowski, T.J.: Natural Patterns of Activity and Long-term Synaptic Plasticity. Current Opinion in Neurobiology 10(2), 172–179 (2000)
13. Izhikevich, E.M.: Weakly Pulse-coupled Oscillators, FM Interactions, Synchronization, and Oscillatory Associative Memory. IEEE Transactions on Neural Networks 10(3), 508–526 (1999)
14. Hoppensteadt, F.C., Izhikevitch, E.M.: Thalamo-Cortical Interactions Modeled by Weakly Connected Oscillators: Could the Brain Use FM Radio Principles? BioSystems 48, 85–94 (1999)
15. Von der Malsburg, C.: Binding in Models of Perception and Brain Function. Current Opinion in Neurobiology 5, 520–526 (1995)

Synchronized Activities among Retinal Ganglion Cells in Response to External Stimuli

Lei Xiao, Ying-Ying Zhang, and Pei-Ji Liang*

Department of Biomedical Engineering, Shanghai Jiao Tong University,
800 Dong-Chuan Road, Shanghai 200240, China
{xiaolei123006,pjliang}@sjtu.edu.cn

Abstract. Synchronized firing is an efficient way for retinal ganglion cells (RGCs) to encode visual stimuli. In the present study, we studied synchronized activities among RGCs in response to natural movie and pseudo-random checker-board flickering. The results showed that nearby RGCs tended to fire synchronously much more frequently than expected by chance, in response to both stimuli. Under our experimental conditions, synchronous groups could contain three or more cells in response to natural movie; but activities were more often observed between pair-wise cells in response to checker-board flickering. The correlation index calculated between neuron pairs did not have any significant tendency of increase or decrease when natural movie stimulation was lasted; however, it tended to increase when pseudo-random checker-board flickering stimulation was lasted.

Keywords: Synchronized activities; correlation index; dynamical; retinal ganglion cells.

1 Introduction

In vertebrates, the optic nerve is a severe bottleneck presented in the visual pathway; dynamic concerted firings are therefore critically required for conveying information effectively [1, 2]. Many lines of evidence from multi-electrode studies of retina have confirmed that adjacent RGCs of similar functional subtype tend to fire in synchrony in response to external stimuli [3-5]. Correlation index, the ratio between the observed concerted firings and that expected by chance, was proposed to quantify the strength of correlation within neuron groups [6].

Over the years, synchronized activities elicited by artificial laboratory stimuli have been studied [4, 6-7], and it was reported that RGCs tend to fire in synchrony more frequently than expected by chance in response to various laboratory stimuli, such as uniform illumination, pseudo-random checker-board stimuli, etc.[6, 8]. However, natural stimuli are usually more complex than artificial stimuli. In order to understand visual function under natural conditions, it is better to study neural responses to natural stimuli directly [9, 10].

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 44–50, 2010.

In the present study, we adopted information-theoretic algorithm [8] to study the dynamically synchronized activities among RGCs in response to natural movie and pseudo-random checker-board flickering stimulation. Correlation index was computed to estimate the strength of synchronous patterns in response to both stimuli. It was found that nearby RGCs tended to fire synchronously more frequently than expected by chance in response to both stimuli. During natural movie, many synchronous groups contained more than three cells; but in response to checker-board flickering; most of groups only contained two cells. For synchronous neuron pairs, correlation index did not show any significant change along with time during natural movie stimulation; but it tended to increase with time in response to pseudo-random checker-board flickering.

2 Materials and Methods

2.1 Electrophysiology Recordings and Visual Stimulation

Detailed extracellular-recording procedure can be found in our previous report [11]. Spikes from RGCs were recorded from retinas of newly-hatched chicks (about 1-3 weeks post-hatching) using multi-electrode array (MEA, 8×8) (MEA60, MCS GmbH, Germany) via a commercial multiplexed data acquisition system with a sampling rate of 20 kHz. Recorded data were stored in PC for off-line analyses.

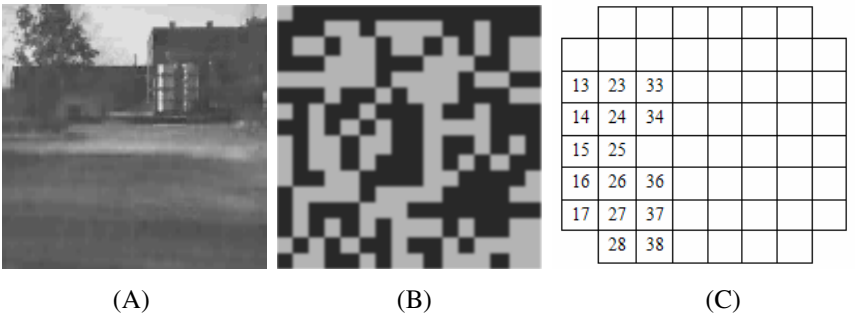


Fig. 1. Example frames and geometric position of eletrodes. (A) Natural movie; (B) Checker-board flickering; (C) Geometric position of 16 adjacent electrodes by which a group of RGCs were recorded from one example retina.

The following stimulation protocols were applied: (1) Full-field white light flashes with light-ON duration of 1 sec and light-OFF intervals of 9 sec were applied (lasted for 30 sec) to test the functional condition of the neurons being recorded; (2) Digitized grayscale video recording of natural movies (downloaded from the website of van Hateren's lab, http://hlab.phys.rug.nl/vidlib/index.html. [12]) were presented with a refresh rate of 10 Hz and lasted for 192 sec; (3) Pseudo-random binary checker-board flickering (16×16 grid) were applied at a refresh rate of 9.05 Hz and lasted for 221 sec [13]. Example frames of natural movie and checker-board flickering are shown in Fig. 1A and B. These images were of the same size when being presented on the screen and projected onto the retinal piece via an optical lens system.

2.2 Information-Theoretic Algorithm

In order to test whether the interactions among ganglion cells are limited to pair-wise neurons or extended to neuron groups containing more cells, information-theoretic algorithm based on entropy analysis was adopted [8]. Detailed procedures are as follows:

Firstly, the spike trains are symbolized into "0" and "1" with time bin of 2 ms, where "1" represents that there is a spike in the time bin and "0" represents that there is no spike in the time bin. Given two neurons *A* and *B*, a new symbolic neuron AB can be defined such that:

$$\begin{aligned} r_j^{(AB)} &= r_j^{(A)} r_j^B \\ &= \begin{cases} 1, \text{ if the neuron } A \text{ and neuron } B \text{ fired in time bin } j \\ 0, \text{ otherwise} \end{cases} \end{aligned} \tag{1}$$

Secondly, to see whether the neurons *A* and *B* are concertedly activated, the entropy is computed:

$$H_i = -(P_i \log_2 P_i + (1-P_i)\log_2(1-P_i)) \tag{2}$$

where P_i is the probability that symbolic neuron i has a spike in the time bin ($P_i = \frac{1}{N}\sum_{i=1}^{N} r_j^{(i)}$, N is the number of time bins in the data set). As for each individual neuron, usually only a small fraction of spikes are fired in synchrony with others, the net reduction in entropy can be calculated as:

$$\begin{aligned} \Delta H_{AB} &= H_A + H_B - H_{AB} \\ &\approx P_{AB} \log_2 (P_{AB} / P_A P_B) \end{aligned} \tag{3}$$

The identification of concerted neuron groups starts with computing ΔH for all the cell pairs. If the largest ΔH value is greater than a predetermined threshold (see below), we regard these two cells as a concerted group. We then further search for other synchronous neuron pairs or synchronous groups containing more cells. The process is repeated until the largest ΔH falls below the predetermined threshold. To define the threshold, all the spike trains are shifted by randomly chosen time delays, and the largest ΔH in the shuffled data set is defined as the threshold.

2.3 Correlation Index

Correlation index is the ratio between the observed frequency of synchronized activities and that expected by chance [6], which is used to estimate the strength of the synchronized firings. The correlation index is measured as follows:

The observed frequency of synchronized firings among *M* cells is:

$$P_{1\ldots M} = \frac{1}{N}\sum_{j=1}^{N}\prod_{i=1}^{M} r_j^{(i)} \tag{4}$$

The frequency of synchronized firings expected by chance can be calculated as:

$$P_1 \ldots P_M = \prod_{i=1}^{M} \frac{1}{N} \sum_{j=1}^{N} r_j^{(i)} \tag{5}$$

Then we can compute the correlation index as:

$$C_{1\ldots M} = P_{1\ldots M} / \prod_{i=1}^{M} P_i \tag{6}$$

3 Results

Most RGCs recorded in our experiments are of On-Off subtype [4], therefore in the present study, analyses were focused on this type of cells. Experiments were performed on 3 retinas. The locations of electrodes by which the neurons' activities were recorded are presented as an approximate indication of the locations of neurons. In the example given in Fig. 1C, the spike trains recorded from 16 adjacent electrodes were analyzed. To reveal dynamically changed population activities among RGCs in response to natural movie and pseudo-random checker-board stimuli, the analyses were performed on the 120-s data sets, and the synchronous groups and strength of the correlation in groups were calculated for each 500-ms period.

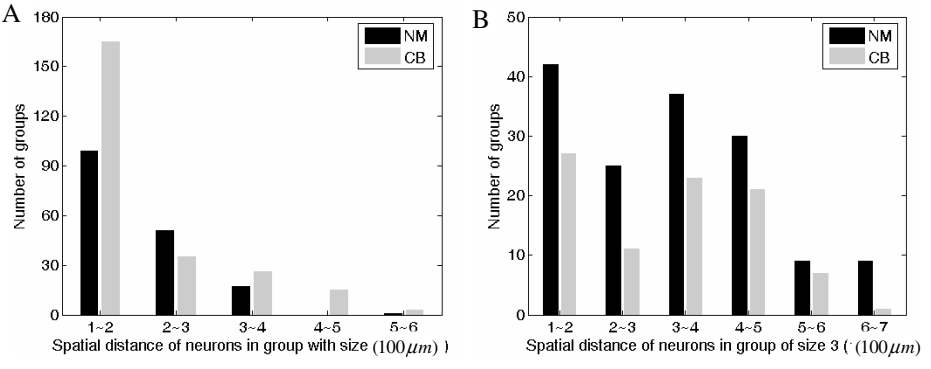


Fig. 2. Spatial arrangement of RGCs engaged in synchronized firing in one example retina (recorded by electrodes presented in Fig. 1C) in response to natural movie (NM) and checker-board (CB) stimuli. A, B. The relationship between number of synchronous groups and inter-neuronal distances, during natural movie (NM) and pseudo-random checker-board (CB) stimulations, with group sizes being 2 and 3, respectively.

3.1 Synchronous Groups

Previous studies have shown that synchronized activities are frequently recorded from adjacent cells [6, 8]. We defined the inter-neuronal distance of a synchronous group using the summation of distances between each neuron and their gravity center.

Fig. 2A (Pairs) and B (Triplet) illustrate the relationship between the inter-neuronal distance and the number of synchronous groups of neurons recorded by electrodes illustrated in Fig. 1C, in response to natural movie and pseudo-random checker-board stimuli from one example retina. It is clear that the number of groups was decreased with distance during both stimuli.

In order to investigate the firing patterns during both stimuli, data collected from three different retinas were analyzed. Table 1 shows the statistic of the group size under various conditions. It is notable that during natural movie, groups could contain three or more neurons; however, most of groups only contain two neurons during pseudo-random checker-board stimuli.

Table 1. Statistic of the number of cells per group

	Stimuli	Pair	Triplet	Quaternion
Retina 1	NM	145	104	22
	CB	242	63	3
Retina 2	NM	179	152	16
	CB	236	90	5
Retina 3	NM	151	145	18
	CB	278	80	3

3.2 Correlation Index

Correlation index represents how frequently a synchronous pattern is observed as compared to that expected by chance [8]. The probability distributions of correlation index for synchronous pairs and synchronous triplets from one example retina (same as presented in Fig. 2) are shown in Fig. 3A and Fig. 3B, for the cells' responses elicited by natural movie and pseudo-random checker-board stimuli. The distributions of synchronous groups in response to both stimuli were similar to each other. Consistent results were observed from the other two retinas.

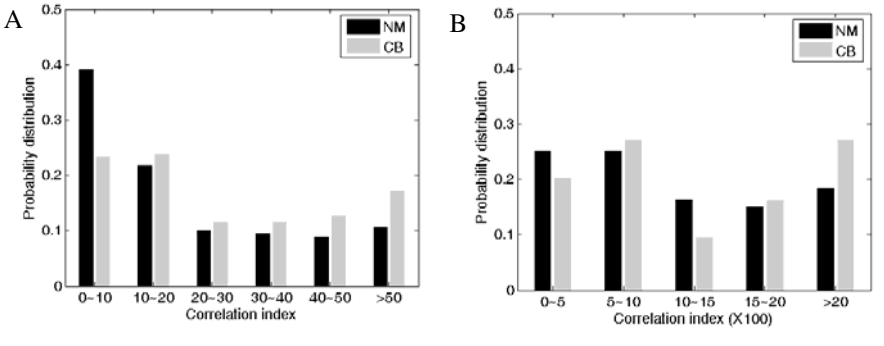


Fig. 3. Distribution of correlation index values from one example retina (the same as presented in Fig. 2) during NM and CB stimuli. A. For synchronous neuron pairs. B. For synchronous neuron triplets.

The results above were obtained by overall analyses perform on 120-s data sets. Actually synchronized activities among RGCs varied dynamically during both stimuli. In order to investigate the time-varying characteristics of correlation index, we analyzed the correlation index of pair-wise neurons by least-squares linear regression fitting. Fig. 4 gives an example pair (#17 and #27 in Fig. 1C). Although the correlation index was fluctuating, it did not have any significant tendency of increase or decrease, during natural movie stimuli (Fig. 4A); but it tended to increase in response to pseudo-random checker-board flicking stimuli (Fig. 4B). The results were observed in almost all the synchronous groups in the three retinas under investigation.

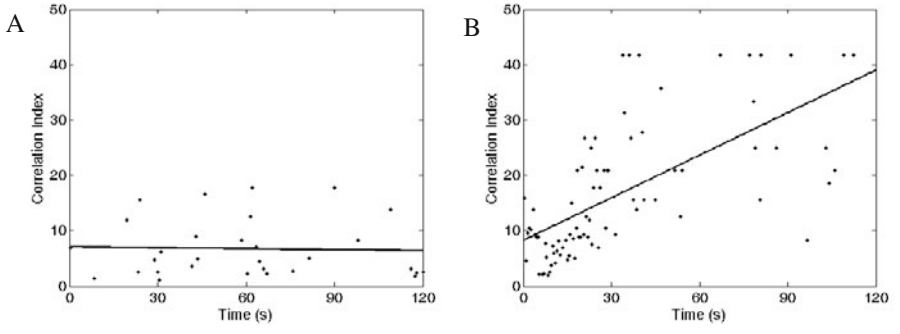


Fig. 4. An example of time-dependent changes of correlation index. The synchronous neurons were recorded by electrodes #17 and #27 in Fig. 1C during both stimuli. A. Duing natural movie stimuli. B. During pseudo-random checker-board stimuli.

4 Discussion

In the present study, we adopt information-theoretic algorithm [8] and correlation index [6] to investigate the concerted activities of neurons recorded by adjacent electrodes in response to natural movie and pseudo-random checker-board stimuli respectively. The results revealed that synchronized activities frequently occurred among adjacent RGCs (Fig. 2). Synchronous patterns elicited by natural movie stimuli were different from that elicited by pseudo-random checker-board stimuli, neurons tended to fire synchronously in larger groups during natural movie stimuli. The distributions of synchronous groups with different correlation index values were almost similar during both stimuli, but the time-varying characteristics of correlation index were very different. For synchronous neuron pairs, the correlation index did not show any significant change along with time in response to natural movie stimuli; however it tended to increase with time in response to pseudo-random checker-board flicking stimuli.

Natural stimuli are fundamentally different from pseudo-random checker-board flicking stimuli in a sense that natural stimuli contain intensive correlations [14, 15] and are spherically asymmetric [15]. It is more frequently that nearby neurons tend to fire synchronously in larger groups during natural movie stimuli. During pseudo-random checker-board stimuli, neurons adapt to the stimuli quickly, which make the

observed frequency of synchronized activities much higher than expected by chance. All of these may suggest that activity patterns among RGCs are different between natural movie and pseudo-random checker-board flicking stimuli and dynamically synchronized activities among RGCs are stronger in response to natural movie.

Acknowledgments. This work was supported by grants from the State Key Basic Research and Development Plan (No.2005CB724301) and National Foundation of Natural Science of China (No. 30670519).

References

1. Meister, M.: Multineuronal codes in retinal signaling. PNAS 93, 609–614 (1996)
2. Usrey, W.M., Reid, R.C.: Synchronous activity in the visual system. Annu. Rev. Physiol. 61, 435–456 (1999)
3. Brivanlou, I.H., Warland, D.K., Meister, M.: Mechanism of Concerted Firing among Retinal Ganglion Cells. Neuron. 20, 527–539 (1998)
4. Liu, X., Zhou, Y., Gong, H.Q., Liang, P.J.: Contribution of the GABAergic pathway(s) to the correlated activities of chicken retinal ganglion cells. Brain Res. 1177, 37–46 (2007)
5. Shlens, J., Rieke, F., Chichilnisky, E.J.: Synchronized firing in the retina. Curr. Opin. Neurobiol. 18, 396–402 (2008)
6. Meister, M., Lagnado, L., Baylor, D.A.: Concerted Signaling by Retinal Ganglion Cells. Science 270, 1207–1210 (1995)
7. Pillow, J.W., Shlens, J., Paninski, L., Sher, A., Litke, A.M., Chichilnisky, E.J., Simoncelli, E.P.: Spatio-temporal correlations and visual signaling in complete neuronal population. Nature 454, 995–999 (2008)
8. Schnitzer, M.J., Meister, M.: Multineuronal Firing Patterns in the Signal from Eye to Brain. Neuron. 37, 499–511 (2003)
9. Touryan, J., Felsen, G., Dan, Y.: Spatial Structure of Complex Cell Receptive Field Measure with Natural Images. Neuron. 45, 781–791 (2005)
10. Lesica, N.A., Jin, J.Z., Weng, C., Yeh, C.I., Butts, D.A., Stanley, G.B., Alonso, J.M.: Adaptation to Stimulus Contrast and Correlations during Natural Visual Stimulation. Neuron. 55, 479–491 (2007)
11. Chen, A.H., Zhou, Y., Gong, H.Q., Liang, P.J.: Luminance adaptation increased the contrast sensitivity of retinal ganglion cells. Neuroreport. 16, 371–375 (2005)
12. Van Hateren, J.H., van der, S.A.: Independent component filters of natural images compared with simple cells in primary visual cortex. Proc. Biol. Sci. 265, 359–366 (1998)
13. Lesica, N.A., Stanley, G.B.: Encoding of natural scene movies by tonic and burst spikes in the lateral geniculate nucleus. J. Neurosci. 24, 10731–10740 (2004)
14. Field, D.J.: Relations between the statistics of natural images and the response properties of cortical cells. J. Opt. Soc. Am. A 4, 2379–2394 (1987)
15. Ruderman, D.L., Bialek, W.: Statistics of natural images: Scaling in the woods. Phys. Rev. Lett. 73, 814–817 (1994)

Novel Method to Discriminate Awaking and Sleep Status in Light of the Power Spectral Density

Lengshi Dai[1], You Wang[1], Haigang Zhu[2], Walter J. Freeman[3], and Guang Li[1]

[1] National Key Lab of Industrial Control Technology, Dept. of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China
guangli@zju.edu.cn
[2] Ningbo Hearty Pamaking Health Mattresses Co. Ltd., Ningbo 315040, China
[3] Department of Molecular & Cell Biology, University of California at Berkeley, Berkeley, CA 94720-3206, USA

Abstract. Sleep stages are mainly classified via electroencephalogram (EEG) which involves some prominent characters not only in amplitude but also in frequency. What is more, researchers are using computer assisted analysis to acquire the panoramic view of long duration sleep EEG. However, unlike the empirical judgment, it is fairly difficult to decide the specific values of sleep rules for computer based analysis due to the fact that those values vary with individuals and different distance from reference to signal source on scalps. This paper will introduce a novel method using power spectral density to discriminate awaking, light sleep and deep sleep EEG just according to the features extracted from EEG.

Keywords: PSD, EEG, Awaking, Light sleep, Deep sleep.

1 Introduction

Since the middle of last century, there are great amounts of scientists doing research on sleep, no matter in pathological field, psychological field or physiological field. And the sleep analysis has been applied into several practical projects such as the diagnosis of epilepsy.

And as a world-wide standard rule, the 'R-K rules' is always being ameliorated since its birth in Washington, 1968. According to this rule, an epoch-by-epoch approach is strongly recommended in all scoring procedures [1]. In the beginning, this rule was just for doctors and researchers to judge the stages by experience; however, the computer assisted skills started to develop since 1980s together with the development of IT industry. While the problem came that for this technology just for manual judgment, the exact value as the standard for analysis was not so clear that the vague borders between stages made it not easy to complete the discrimination correctly, resulting in the time-consuming process of determining the specific parameters. As three sorts of typical sleep stages, the aim of this paper is to distinguish the states of awaking status, light sleep and deep sleep using power spectral density. We have got to know that the background activity of human body includes black noise (i.e.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 51–57, 2010.

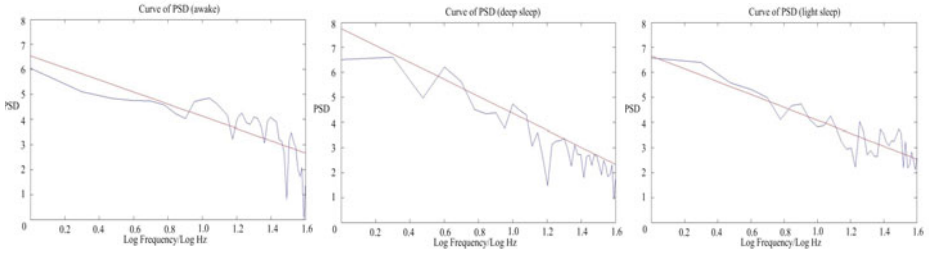
background noise mainly in low frequency), brown noise (i.e. background noise mainly in low and middle frequency) and pink noise (i.e. background noise mainly in relatively higher frequency). The background activity of cerebral cortex in states of rest and slow wave sleep resembles broadband noise. The power spectral density (PSD) then may often conform to a power-law distribution: a straight line in coordinates of log power vs. log frequency [2]. For human race, in slow wave sleep (SWS) PSD decreases at $1/f^{\alpha}$, α~3,with loss of beta–gamma spectral peaks and diminished or absent oscillations and spatiotemporal phase structure. In the awaking state, the power spectral density (PSD) showed power-law decrease in log power with log frequency at $1/f^{\alpha}$, α~2, but with peaks in the standard empirical ranges [3]. In short we are able to conclude that for the state of rest, it resembles the brown noise with its slope approximately close to -2, and for deep sleep, it decreases to the range from -3 to -4 which resembles the black noise, while for light sleep, it locates at the intermediate site of the former two, and these persuasive background activities are caused by mutual excitation among cortical neurons [4]. The following contents will introduce the method to extract the distinction with PSD.

2 Methods

We choose ten blocks of one-channel EEG signal with each block consists of five-minute waking EEG, five-minute light sleep EEG (mainly in stage II for stage I is only a transient state and is not representative enough) and five-minute deep sleep EEG. These segments of EEG signal are from the dataset which was built for sleep quality analysis. It was digitized at 250Hz and with low pass filter from 0-40Hz. The data has been preliminarily judged by the method combined with Hilbert Huang Transform (HHT) and 'R-K rules' where the HHT algorithm acts as an effective way to acquire the frequency spectrum and R-K rules as the criterion for judgment, first the frequency spectrum is calculated by HHT, then these results combined with the amplitudes of both EEG and EOG are applied into the R-K rules which would determine the specific sleeping stage of each segment [5]. According to 'R-K rules', the most prominent character of awaking status is that the EEG contains alpha activity and/or low voltage, mixed frequency activity. The light sleep contains fewer alpha activities and is characterized as the spindle waves and K-complex waves; for deep sleep it elaborates that it contains great amounts of high amplitude, slow wave activity [1].

For each block, we just divided the three five-minute signals into 300 segments separately with each segments 1 second (for the sampling frequency is 250Hz, signal for each second consist of 250 points). Then we calculated the frequency spectrum of each segment in use of FFT; hence the log frequency vs. log power figure comes out. Meanwhile, we use first order fitting to build a linear function in form of y=b*x+c where the parameter 'b' represents the slope of the fitting curve and c for the intercept. (Fig.1)

In Fig.1(a), Fig.1(b), Fig.1(c) it is obvious that for deep sleep stage, the power in low frequency bands like delta is overwhelmingly stronger than that of the waking stage (one order stronger), while for the relatively higher frequency bands like alpha and beta, the waking stage contains more power than sleep stage, that is why the sleep



(a) (b) (c)

Fig. 1. The PSD of one second EEG signal (the blue curve) and the straight fitted to PSD(the red curve), this PSD is from a random one second of different status. (a) awake, (b) deep sleep and (c) light sleep.

signal approximates to black noise which is characterized by the slope below -3 and the waking signal approximates to brown noise with its slope around -2. In contrast, the light sleep has relatively strong power in lower frequency compared to that of awaking status and stronger in power of 10-16Hz than that of deep sleep, determining the slope of this stage around -2.5. This is due to the K-complex activity which is in lower band (i.e. less than 2Hz) and the spindle waves which has a higher frequency band (i.e. 10-16Hz) All those above are coherent to the description in R-K rules that in waking stage, it is mainly occupied with alpha activity for more than 50% of the whole epoch in temporal domain; in deep sleep, the delta wave with peak to peak amplitude more than 75uv dominates more than half of the EEG signal in temporal domain; in light sleep, there is less alpha activity than awaking status and dominated by spindle waves and K-complex waves. This is also the cause of the steeper slope in the PDS of deep sleep.

With the acquirement of these PSD graphs, it is now possible to extract some features; thus we decide to utilize several parameters to express the characters of each segment of EEG signal. Other than the slope and intercept is the parameter named SI, which is defined as

$$SI = \frac{\sum_{n=1}^{40} |\log_{10} p| - \sum_{n=1}^{40} |\log_{10} R|}{\sum_{n=1}^{40} |\log_{10} p|} \tag{1}$$

Where

$$\sum_{n-1}^{40} |\log_{10} R| = |\sum_{n=1}^{40} (\log_{10} p - \log_{10} p_e)| \tag{2}$$

Here $\log_{10}p_e$ is the simulated value in the fitting curve. It expressed the percentage of the total variance in the designated frequency range, the smaller this index is, the more intense the frequency ranges, and the more active the state of brain is. In addition, to express the activity of alpha band which involves the prominent character of

waking state, we use the SIα, which is the value of SI specifically in the alpha band (8-13 Hz).

$$SI\alpha = \frac{\sum_{n=8}^{13} |\log_{10} p| - \sum_{n=8}^{13} |\log_{10} R|}{\sum_{n=8}^{13} |\log_{10} p|} \tag{3}$$

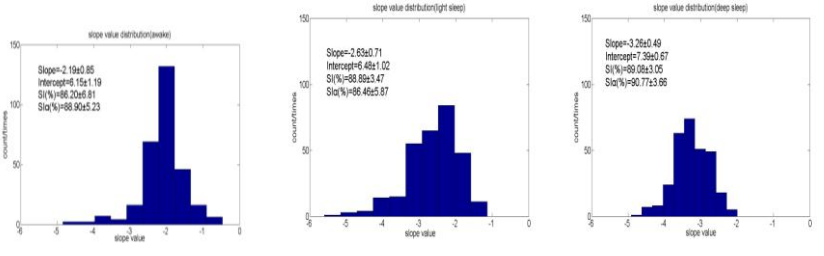
And

$$\sum_{n-8}^{13} |\log_{10} R| = |\sum_{n=8}^{13} (\log_{10} p - \log_{10} p_e)| \tag{4}$$

Also, to avoid the relatively small number of discrete point in PSD figures (totally only 40 points in log frequency vs. log power coordinate) which may not agree with law of large numbers and would create some uncertainty in calculation, it is necessary to interpolate additional points into the PSD curves. We just use spline as the way to realize the interpolation, more than 1600 points for each function are interpolated into the PSD and the fitting curve with the step as 0.001.

3 Results

The 300 segments vided in each five-minute EEG signal make it necessary to do some statistical works especially for the slope which represents the typical distinction among deep sleep, awaking status and light sleep. (Fig.2)



(a) (b) (c)

Fig. 2. Hisrograms of the slopes of PSD from human EEG (a) awake, (b) deep sleep, and (c) light sleep

The histogram in Fig.2 (a) indicates the slope near -2 and resembles brown noise. In addition, there is the tail downside towards the slope of -5. These sporadic deviations show the tendency from brown noise to black noise and the unsteady state of the waking stage. That means in this block, the waking stage goes unstable compared to deep sleep. The distribution of Fig.2 (b) appears similar to symmetry distribution,

with the average slope between -3 and -4, which resembles the black noise. In Fig.2 (c), the average value of slopes is around -2.5 which goes in the middle of the waking stage and deep sleep stage with a tail towards lower value, this is because of the random occurrence of K- complex waves which has peak to peak amplitude more than 100uv, and together with the fluctuation of spindle waves in 10-16 Hz, it appears to be lower in SI and SI α.

In sum, the results are detailed in Table 1 Table 2 and Table 3, those assert that the slope of waking stage is around -2 and for deep sleep, it descends to below -3 with higher value for intercept, the slope of light sleep is in medial. Meanwhile, it also states that no matter for SI or for SIα, EEG in awake and rest is a little bit smaller than that of deep sleep for around two percentiles which means the waking EEG fluctuates more actively than sleep EEG, both in the alpha band or in the overall frequency. While for light sleep these two parameters are often lower maybe due to the fluctuation in both lower and higher frequency bands.

Table 1. The specific details for waking EEG

	Deep sleep (average±SD)			
Index / No	**SI (%)**	**SIα (%)**	**Slope**	**Intercept**
1	86.96±5.00	89.15±4.24	-3.26±0.59	7.23±0.83
2	87.25±3.65	90.04±3.23	-3.55±0.44	7.35±0.58
3	87.28±4.60	89.98±3.90	-3.32±0.60	7.58±0.82
4	88.40±4.26	90.35±3.64	-3.43±0.50	7.65±0.71
5	88.26±4.03	89.90±3.79	-3.53±0.50	7.69±0.72
6	89.30±3.15	90.93±2.90	-3.47±0.47	7.58±0.64
7	88.59±3.41	90.20±2.92	-3.42±0.48	7.64±0.65
8	89.08±3.05	90.77±3.66	-3.26±0.49	7.39±0.67
9	87.30±3.86	90.15±3.43	-3.67±0.51	7.85±0.70
10	87.94±3.96	90.08±3.45	-3.29±0.40	7.06±0.53

Table 2. The specific details for sleep EEG

	Wake (average±SD)			
Index / No	**SI (%)**	**SIα (%)**	**Slope**	**Intercept**
1	85.69±7.02	87.46±6.46	-1.64±0.78	5.35±1.12
2	86.82±5.92	89.05±5.17	-1.87±0.89	5.47±1.42
3	86.74±5.87	89.02±4.90	-1.82±0.84	5.82±1.12
4	86.38±7.03	89.02±6.06	-1.51±1.34	5.60±2.03
5	86.78±5.05	89.02±4.90	-1.61±0.63	5.55±0.96
6	86.71±5.83	87.82±5.43	-2.17±0.76	6.19±0.99
7	86.71±5.64	88.77±4.70	-2.05±0.86	6.11±1.14
8	86.20±6.81	88.90±5.23	-2.19±0.85	6.15±1.19
9	87.82±4.30	89.50±3.78	-2.25±0.50	6.11±0.69
10	86.38±8.16	88.54±8.05	-1.83±1.67	5.77±2.64

Table 3. The specific details for sleep EEG

	Light sleep (average±SD)			
Index / No	**SI (%)**	**SIα (%)**	**Slope**	**Intercept**
1	82.91±8.98	86.85±6.45	-3.15±1.22	7.21±1.78
2	84.67±6.15	88.96±4.69	-3.12±0.75	7.01±1.08
3	82.48±10.45	86.14±8.20	-3.08±1.40	7.50±2.05
4	83.02±10.80	85.79±8.64	-3.09±1.42	7.46±2.07
5	84.83±8.65	87.70±6.62	-3.02±1.17	7.25±1.69
6	86.76±5.06	89.28±4.12	-2.58±0.65	6.61±0.92
7	85.67±6.93	88.88±4.36	-2.86±0.81	6.90±1.13
8	86.46±5.87	88.89±3.47	-2.63±0.71	6.48±1.02
9	82.72±7.32	88.10±4.79	-3.40±0.90	7.50±1.30
10	87.02±5.47	89.75±4.28	-2.44±0.59	6.38±6.38

We also notice that the standard deviation in deep sleep is usually much smaller than those in the waking status with rare exceptions. We induce that it could also demonstrate that in deep sleep, the brain activity is more stable than waking periods. Another reason why the slope of waking status deviates a bit from the theoretical value -2 is due to the muscle activity (i.e. EMG) mingled into the waking EEG and forced the slopes closer to pink noise which involves higher frequency bands.

3 Discussion

Based on the power spectral density, we may find the regular patterns that for waking EEG, it is similar as brown noise, and for deep sleep EEG it is closer to black noise whose slope is much steeper, but for light sleep it is in the medial of those two; and the difference in SI and histograms shows the relatively unstable nature of waking EEG and light sleep with more fluctuations and sporadic deviations. The lower value of SIα in waking stage and light sleep just corresponds to the R-K rules that the alpha activity usually occurs in Stage W and spindle occurs in light sleep.

Hence we can conclude the radical methods in PSD based sleep stage classification, from higher to lower in absolute value, it is deep sleep, light sleep and awaking status for the value of slope; deep sleep, awaking status and light sleep for the value of SI and SIα; deep sleep, light sleep and awaking status for value of intercept. Compared to the traditional computer based time-frequency analysis, the method of PSD has its distinctive advantages. Unlike the time-frequency analysis, this method would no longer be restricted by the specific details of R-K rules, which means the distinctive difference between slopes, intercepts and SI would help to overcome the fuzzy boundary between sleep stages. Additionally, this method could be utilized for different individuals without changing the parameter of the algorithm for it is adaptive to different subjects where the comparison would occur only in the same platforms; unlike neural networks, there is no need to do any training works in advance to adjust the specific parameters and inner configurations. Accordingly, the method of PSD is also the allusion of the mechanism of human's brain, not only for its general condition

of stability but also for transition of background noise. It is probable to say that this approach is another perspective to express the brain activity described in R-K rules and an effective way to realize sleep stage classification.

Acknowledgements

The work is supported by the National Natural Science Foundation of China (Nos. 60874098 and 60911130129), the High-Tech Research and Development Program (863) of China (No. 2007AA042103) and the National Creative Research Groups Science Foundation of China (No. 60721062).

References

1. Rechtschaffen, A., Kales, A.: A Manual of Standardized Terminology, Techniques and Scoring System for Sleep Stages of Human Subjects. Brain Inform. Service/Brain Res. Inst. (1968)
2. Freeman, W., Zhai, J.: Simulated power spectral density (PSD) of background electrocorticogram (ECoG). Cognitive Neurodynamics 3(1), 97–103 (2009)
3. Freeman, W., Holmes, M.D., West, G.A., Vanhatalo, S.: Fine spatiotemporal structure of phase in human intracranial EEG. Clinical Neurophysiology 117, 1228–1243 (2006)
4. Freeman, W.: Definition of state variables and state space for brain-computer interface. Cognitive Neurodynamics 1, 3–14 (2007)
5. Dai, L., Dai, G., Zhang, W., Zhu, H., Li, G.: Sleep quality analysis based on HHT. In: The Second International Conference on Cognitive Neurodynamics, Hangzhou (2009)

Current Perception Threshold Measurement via Single Channel Electroencephalogram Based on Confidence Algorithm

You Wang[1], Yi Qiu[1], Yuping Miao[2], Guiping Dai[3], and Guang Li[1]

[1] National Key Lab of Industrial Control Technology, Dept. of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China
guangli@zju.edu.cn
[2] Hangzhou Prospect Science and Technology Co. Ltd., Hangzhou 310003, China
[3] Zhejiang Food and Drug Administration, Hangzhou 310012, China

Abstract. A new algorithm for pattern recognition for current perception threshold measurement is proposed. Because of risk-sensitiveness of medical diagnosis, the current perception threshold requires the prediction to be qualified with confidence. Using confidence support vector machine, the current perception threshold measurement via single channel electroencephalogram is confident and credible. Experimental results demonstrate the proposed algorithm is effective.

Keywords: Current Perception Threshold (CPT), Transductive Confidence Machine (TCM), Support Vector Machine (SVM), Electroencephalogram (EEG), Quantitative Sensory Testing (QST).

1 Introduction

Current perception threshold measurement is a threshold detection of quantitative sensory testing (QST) which have been developed to assess and quantify neural sensory function in patients at risk of developing neurologic disease or with neurologic symptoms [1]. In recent years, there has been an increasing interest in study of the relationship between EEG patterns and QST detection, which help us understand the mechanism of neural information processing and provides methods for neural sensory function estimation and brain-computer interface (BCI) applications [2]. Utilizing event related potentials (ERPs), we develop an objective and non-intrusive neural sensory function measurement method, which can identify current perception threshold via single channel EEG. Because medical diagnosis such as CPT measurement is risk-sensitive, the algorithm for pattern recognition requires be qualified with confidence and credibility. Transductive confidence machine (TCM) allows us to make predictions with confidence, which can provide a calibrated and controlled classification environment [3]. We implement TCM on support vector machines (SVM) for EEG pattern recognition in our research.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 58–62, 2010.

2 Methods

The procedure to EEG pattern recognition evoked by current stimuli is illustrated in Fig.1. Because principal energy of ERPs evoked by current stimuli is dominant in the frequency band of 0-3Hz, the single channel EEG was firstly filtered with a low pass filter of 0-3Hz, then segment properly. Secondly, we calculated the Euclidean distances between the segmented EEG and the EEG templets with and without stimuli. The Euclidean distances form a 2-dimension feature vector indicated as X_1 and X_2 respectively. Applied TCM-SVM algorithm, current perception threshold can be obtained.

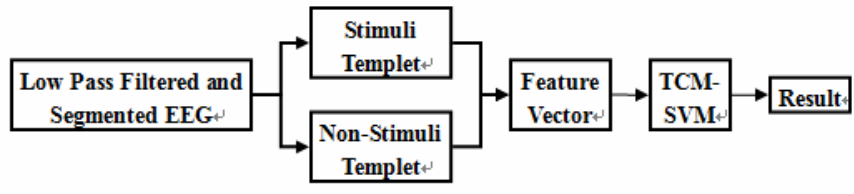


Fig. 1. The procedure to EEG pattern recognition utilizing TCM-SVM

2.1 Experiments

Three male students were selected from right-handed volunteers, who were free of neurological or psychiatric disorders. The subjects seated on a comfortable chair, eyes closed and arms rested on the table during the experiments. When the electrical current was applied on the index finger's end joint of the subject's left hand with a stimulating electrode, monopolar EEG data were acquired at a sampling frequency of 500Hz using NuAmps Digital Amplifier (Model 7181) purchased from the Neuroscan Compumedics Limited, Texas, USA, according to the international 10-20 system. At the same time a synchronous signal was sent to the amplifier by the electrical current generator. In practical applications, fewer EEG recording channels are preferred, at the same time it happens in different parts of the scalp corresponding to the different stimulus. In our experiment there's tiny difference between single channel EEG and multi channels EEG. According to ERPs evoked by electrical stimulus are the most obvious in somatosensory part. The CPz channel EEG selected for exploration was recorded, which referenced to linked mastoids with a forehead ground, and all impedances were kept below 5 Kohm during recording.

The ERPs appeared at the time of about 600ms after current stimulus, however when the current is low there were not ERPs. According to the ERPs analysis, we extracted 128 samples around the peak of the ERPs from single channel EEG signal, and calculated the Euclidean distances to stimulating and non-stimulating EEG templets respectively [2]. A 2-dimensional feature vector was obtained. Two-class SVM classification used for current perception threshold was illustrated in Fig.3. [4]

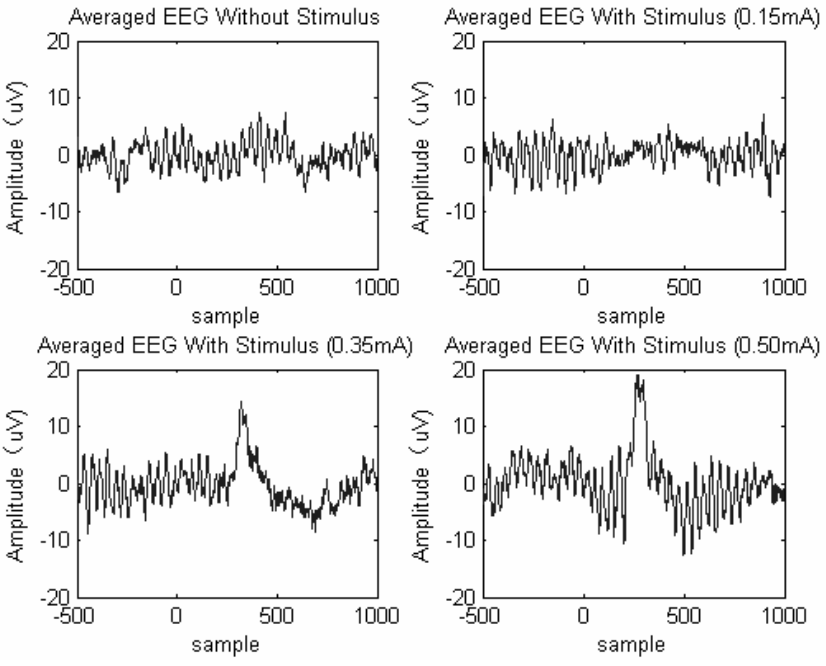


Fig. 2. Average EEG signal with electrical current of different amplitude

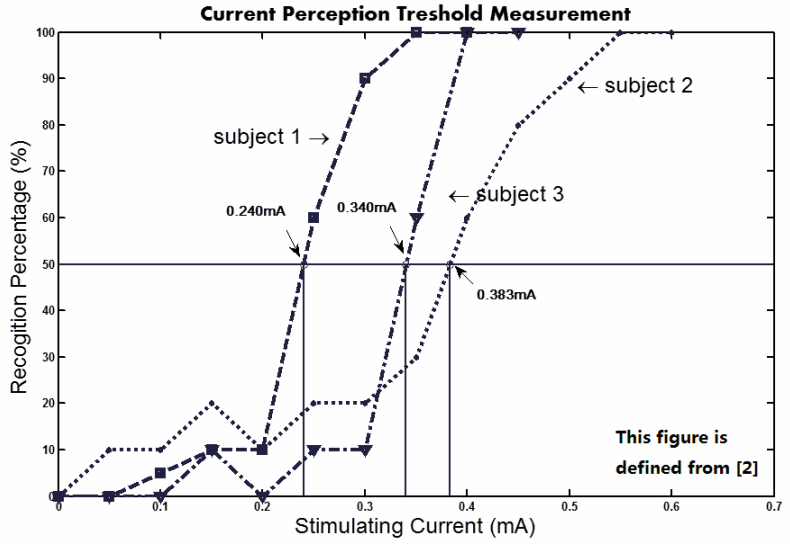


Fig. 3. This figure shows the percentage of classifications. The recognition percentage represents the patterns classified as current perception ERPs.

Fig. 3 represents the relationship between EEG patterns and the amplitude of electrical current. The subjects' ERPs were detected less when electrical current is weak, whereas ERPs can be detected more if strong current. Although the SVM classification algorithm can only give bare predictions, current perception threshold can be obtained statistically, 0.240mA, 0.383mA and 0.340mA respectively for all theses subjects. This method has certain disadvantages, such as: The algorithm for pattern recognition gives "bare" predictions only, lacking reliability and confidence for every prediction made. Experiments need too many stimuli for statistical analysis, so the perception of subject's neural sensor gets weak for excessive stimuli. So we must propose a new method which provides confidence and credibility for predictions and is practical in an efficient manner.

2.2 Transductive Confidence Machines

Transductive confidence machines are supervised machine learning algorithms, which make a prediction regarding a new object based on a training set based on the so called algorithmic theory of randomness. Our algorithm follows the transductive approach, as for the classification of every new example it uses the whole training set to infer a rule for that particular example only. The only assumption is that data sets are produced independently by the same stochastic mechanism (independent and identically distributed assumption, iid assumption). The iid assumption is a very natural one for most applications of pattern recognition [5]. In confidence machines, the problem of prediction with confidence is formalized as that of computing "predictive regions" rather than bare predictions. We set a confidence level. For each new example, we predict the set of labels. In current perception threshold measurement, a sequence of training examples of objects X_i with labels Y_i, where X_i are two-dimensional vectors

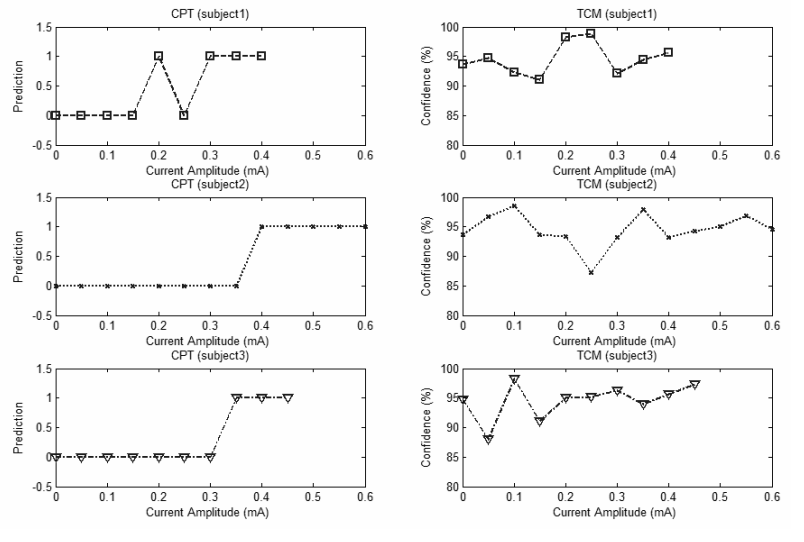


Fig. 4. TCM-SVM algorithm for current perception threshold measurement. Left figures shows prediction made by TCM-SVM, 0 represents label of non-perception and 1 represents label of perception. Confidence of classification is in right figures.

and Y_i are classification taking only finite number of values such as {non-perception, perception}, for i = 1 to n – 1, is given, then a prediction for a new unlabelled object X_n is made. In the TCM, p-values are calculated for each possible label prediction Y_n = Y, based on "strangeness values" a_i that are calculated for each example and indicate how "strange", or how typical. For the implementations of TCM on SVM, we use the Lagrangians a_i obtained in the dual setting of SVM as strangeness values.

2.3 Results

TCM-SVM was applied to the current perception threshold measurement, which can be obtained, 0.225mA, 0.375mA and 0.325mA respectively for all theses subjects illustrated in Fig. 3.

TCM makes predictions within predefined confidence levels, 85%, thus providing a calibrated and controlled classification environment.

3 Conclusions

An EEG-based current perception threshold measurement using a two-class SVM with transductive confidence machine algorithm has been presented. Efficient algorithm of TCM-SVM can reduce the experiment stimulating count, and makes prediction within predefined confidence levels. Current perception threshold has been obtained objectively.

Acknowledgements

Project supported by the National Natural Science Foundation of China (Nos. 60874098 and 60911130129), the High-Tech Research and Development Program (863) of China (No. 2007AA042103) and the National Creative Research Groups Science Foundation of China (No. 60721062).

References

1. Shy, M.E., Frohman, E.M.: Quantitative Sensory Testing. American Academy of Neurology 60, 898–904 (2003)
2. Qiu, Y., Li, G.: Single-Trial Electrical Perception Recognition via Single-Channel Electroencephalogram. In: The 2nd International Conference on Cognitive Neurodynamics, Hangzhou (2009)
3. Luo, Z., Bellotti, T., Gammerman, A.: Qualified Predictions for Proteomics Pattern Diagnostics with Confidence Machines. In: Yang, Z.R., Yin, H., Everson, R.M. (eds.) IDEAL 2004. LNCS, vol. 3177, pp. 46–51. Springer, Heidelberg (2004)
4. Miao, Y., Dai, G., Qiu, Y., Dai, L., Li, G.: Recognition of Current Perception Based on Single-trial Single-channel EEG Analysis. Chinese Journal of Sensors and Actuators 23, 1–4 (2010)
5. Proedrou, K., Nouretdinov, I., Vovk, V., Gammerman, A.: Transductive Confidence Machines for Pattern Recognition. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) ECML 2002. LNCS (LNAI), vol. 2430, pp. 381–390. Springer, Heidelberg (2002)

Electroantennogram Obtained from Honeybee Antennae for Odor Detection

You Wang[1], Yuanzhe Zheng[1], Zhiyuan Luo[2], and Guang Li[1]

[1] National Key Lab of Industrial Control Technology, Department of Control Science and Engineering, ZheJiang University, Hang Zhou, 310027, China
guangli@zju.edu.cn
[2] Computer Learning Research Centre, Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK

Abstract. Electroantennogram and spiking trains were obtained from honeybees' antennae for detection using an apparatus described in this paper. Different odor stimuli were applied on the antennae and got different EAG responses. The spikes of neurons were observed with this system too.

Keywords: Electroantennogram; EGA; Insect antenna; Honeybee; Stimulus.

1 Introduction

Gas detecting technology has a variety of application, the researches main focus on the gas sensors. But most of them are not sensitive enough, and it may take a long time to be as sensitive as a dog or an insect. Insects' antennae are very sensitive, can diagnose the tiny odors within a complex chemical background. For example, Honeybees had already been used in the airport as detectors to search the explosive. So the scientists begin to extract the signals from the antennae of the honeybees with the electrodes [1][2].

The sum of the responses of thousands of olfactory receptor neurons on insect antennae comprise the whole antennal response to odor is named as the electroantennogram (EAG). Ever since it was invented by Schneider in 1957, EAG recording has become one important techniques in insect olfaction and pheromone bioassay research [3]. After olfactory training, the electrophysiological data reveal that training to different odors induced different effects on the antenna sensitivity. These effects were found only in the learners' group [4].

The goal in this study was to build up an EAG detecting system to detect the different response of the honeybees to different stimulus for further odor discrimination or recognition.

2 Experiment

2.1 Experimental Animals

A Chinese honeybees (Apis cerana) hive is kept in the institute, and feed with honey in the winter. The adult drones were collected from the top of the hive or outside of the hive, and kept in Pasteur pipettes.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 63–66, 2010.

2.2 Set-Ups and Operation

The set-ups consists of 4 parts as shown in Fig.1: micro-manipulators, a differential pre-amplifier (ADA400A, Tektronix Inc., USA), a stimulus application system and an oscilloscope (TDS5054B, Tektronix Inc., USA). The diagram of the differential pre-amplifier is shown in Fig.2. Artifacts caused by background signals in the immobilized bees were greatly reduced by thoroughly designed faraday cage. The antennae were cut off from drone together with its head using micro-scissors, to keep the antennae to be alive for longer time. Clip off a few segment from the tip of the antennas. On the micro-manipulators, antennae were connected to one Ag/AgCl recording electrode by the electrically conductive gel and the head was connected to another Ag/AgCl electrode as the reference. If the resistance between the two electrodes was below 2 MOhm, the contact of the electrodes was sufficient. The surface of the manipulator is covered by the gold film to increase the connection. We used multiple antennae in parallel to improve signal-to-noise ratio in EAG responses [5]. There were two tubes in the stimulus set-up; each was connected with one conical flask, and two tubes join together to be a mix-odor tube. The mix-odor tube is directly face to the antenna, the distance between them is about 1cm. A flow meter was adapted to measure and control the flow. The flow was usually adjusted about 0.6L/min. Use a switch pedal to apply test puff. The sample rate of the oscillator was set as 5K sample/s to continue recording 40s by one channel, while the other channel was used to mark the starting and the end time of the stimulus synchronously. Up to now, two kinds of the gas were used to stimulate the antenna in the experiments. One is saturated honey vapor in the top of the conical flask, and the other is water vapor. The gas kind can be easily extended by adding the conical flasks.

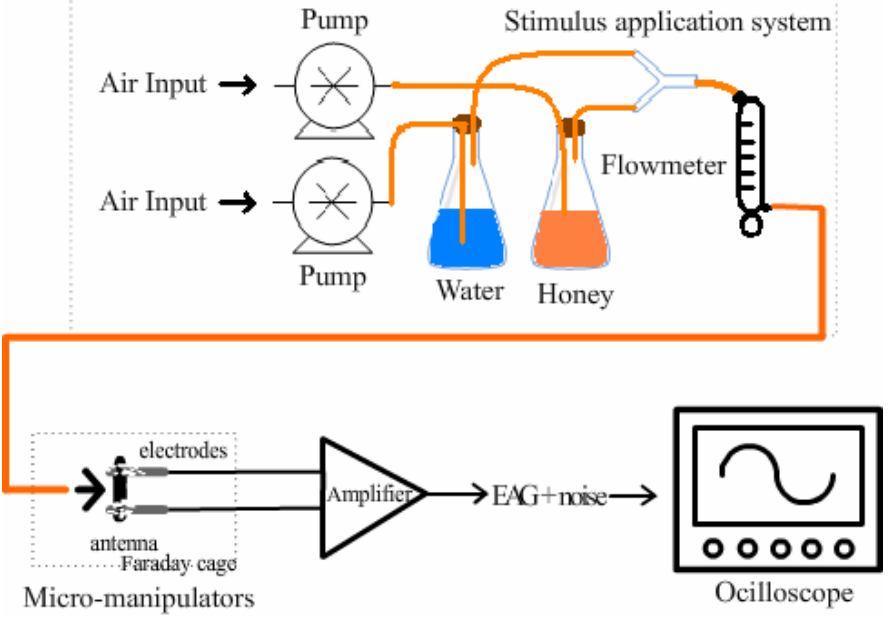


Fig. 1. Experimental apparatus

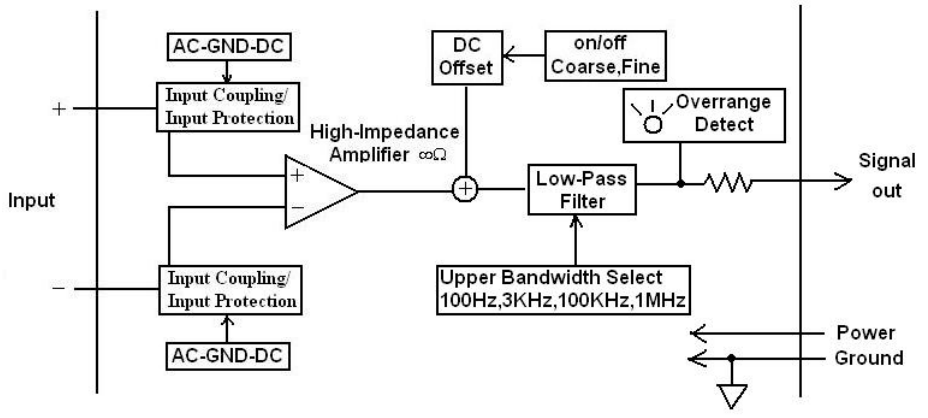
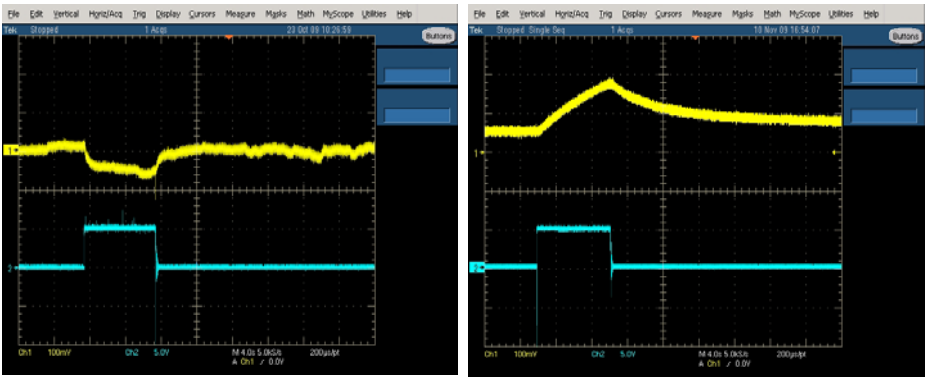


Fig. 2. Diagram of the Differential Preamplifier

3 Results and Discussion

Typical EAGs corresponding to water and honey vapors are shown in Fig.3. The blue line marks the duration of the stimulus, while the yellow line (above) is the response of the antenna to the different stimuli. Without any training on the honeybee, the EAG responses were not guaranteed when a stimulus applied. The rate of obtaining EAG is 49.5% for honey vapor, and 94.1% for the water vapor.

Besides the EAG waveforms, the spiking trains of the antenna as shown in Fig.4. They may provide more information for odor discrimination for further study.



(a) (b)

Fig. 3. (a)The EAG of the honey vapor (b) The EAG of the water vapor. The blue lines (below) mark the start and the end of the stimulus, the yellow lines (above) are the response of the antennae to different stimuli.

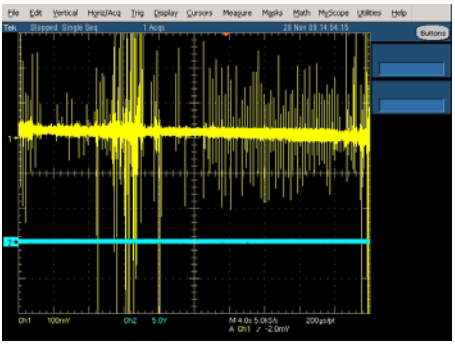


Fig. 4. The spiking train of an antenna

4 Summary and Discussions

In order to utilize insect antenna as an ultra-sensitive sensor for odor discrimination, an EAG measurement system was built up. Based on the set-up, the EAG signals of the honeybees' antennae corresponding to different odor stimuli were recorded. Based on the EAG datasets collected, the experiments for odor discrimination will be carried out.

The spiking trains of the honeybee antenna were obtained by carefully adjusting appropriate parameters of the system. There should be more detailed odor information hidden within them in comparison with EAG. Hopefully, the spiking trains corresponding to different odor stimuli will provide efficient information for odor recognition.

Acknowledgements

The work is supported by the Natural Science Foundation of China (Grants No. 60874098 and 60911130129) and the National Creative Research Groups Science Foundation of China (NCRGSFC: 60721062).

References

[1] Barsan, N., Koziej, D., Weimar, U.: Metal oxide-based gas sensor research: How to? Sensors and Actuators B: Chemical 121, 18–35 (2007)
[2] Harris, S.: The honey trap, Engineering & Technology, pp. 24–26 (December 2007)
[3] Schneider, D.: Elektrophysiologische Untersuchungen von Chemound Mechanoreceptoren de Antenne des Seidenspinners Bombyx mori I. Zeitsehrift fiir vergleiehende Physiologie 40, S8–S41 (1957)
[4] Jong, R.D., Pham-Delègue, M.-H.: Electroantennogram responses related to olfactory conditioning in the honey bee. Journal of Insect Physiology 37, 319–324 (1991)
[5] Park, K.C., Baker, T.C.: Improvement of signal-to-noise ratio in electroantennogram responses using multiple insect antennae. Journal of Insect Physiology 48, 1139–1145 (2002)

A Possible Mechanism for Controlling Timing Representation in the Cerebellar Cortex

Takeru Honda[1,2], Tadashi Yamazaki[3], Shigeru Tanaka[1,2], and Tetsuro Nishino[1]

[1] Faculty of Electro-Communications, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu-shi, Tokyo 182-8585, Japan
[2] RIKEN Brain Science Institute, 2-1 Hirosawa, Wako-shi, Saitama 351-0198, Japan
[3] RIKEN BSI-TOYOTA Collaboration Center, 2-1 Hirosawa, Wako-shi, Saitama 351-0198, Japan
{takeru,shigeru,nishino}@ice.uec.ac.jp, tyam@brain.riken.jp

Abstract. We have developed a network model of cerebellar cortex, in which granular cells' activities represent a passage of time from the onset of a conditioned stimulus (CS). Long-term depression of parallel fiber synapses at Purkinje cells (PCs) encodes an interstimulus interval between onsets of a CS and an unconditioned stimulus (US) as cessation of PC firing, resulting in the emission of a conditioned response (CR) from cerebellar nucleus neurons. In this study, we show that a change in the strength of a CS extends or compresses spike trains of granule cells in the time dimension, suggesting controllability of CR timings flexibly after conditioning. Because PCs alone are insufficient to read out a modified interstimulus interval, we add stellate cells (SCs) inhibiting PCs. Thereby, after conditioning, PCs are shown to stop firing earlier or later than the US timing for a CS stronger or weaker than the CS during conditioning.

Keywords: Cerebellum, Spiking network model, Passage of time, Eyeblink conditioning, Adaptive timing.

1 Introduction

The cerebellum is widely accepted to control the performance of motor actions precisely [1]. Achievement of temporally coordinated motor actions requires precise timing of each action. Very recently, we have proposed that neural dynamics of the granular layer in the cerebellar cortex represents the passage of time (POT) from the onset of a sustained conditioned stimulus (CS) conveyed by mossy fibers (MFs) as a temporal sequence of active granule cell (GRC) populations [2]. Combining with long-term depression (LTD) at synaptic junctions of parallel fibers (PFs) from GRCs to PCs, which is induced by repetitive pairings with a CS and an unconditioned stimulus (US) conveyed by the climbing fibers, PCs stop firing at the timing of the US presented during conditioning. The cessation of the PC firing releases tonic inhibition to cerebellar nucleus neurons that receive direct MF signals, and elicits a conditioned response (CR) as the nucleus neurons' activation.

When we apply these computational mechanisms to the conditioning of motor reflex, it can be accounted for the temporal topography of a CR to a CS presentation.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 67–76, 2010.

A typical example of such conditioning is Pavlovian delay eyeblink conditioning (for review, e.g., [3], [4], [5], [6]). Particularly, in this conditioning, a subject is exposed to paired presentation of a sustained tone as a CS and an airpuff as a US that induces eyeblink reflex. After the repeated conditioning by CS-US presentation with a fixed interstimulus interval (ISI) between the CS and US onsets, the CS presentation alone let the subject close the eye as a CR slightly prior to the US onset timing.

In our simulations of a conditioned cerebellar network, we found that spike trains elicited by GRCs were extended or compressed for a CS stronger or weaker than the CS during the conditioning, which indicates that a stronger CS let a POT shorter whereas a weaker CS let it longer. This possibility is suggested by a recent experiment, in which a CR was elicited earlier when the strength of a tonal stimulus (CS) was increased. However, in the simulations, when we changed the strength of a CS, a model PC did not stop firing clearly. This suggests that the PC's read-out mechanism of a POT represented by GRCs is not sufficient. In the present study, we incorporate a stellate cell (SC) that inhibits the PC. We assumed that the model SC receives climbing fiber inputs and CS-US conjunctive stimulation induces long-term potentiation (LTP). Simulations showed that the model PC successfully stop firing earlier or later than the timing of a CR elicited by the CS presented during conditioning. It is suggested that SCs work to assist and enhance the PC's read-out mechanism of a POT.

2 Methods

We build our model based on the GENESIS script of the granular layer model, which was written by Maex and De Schutter [7]. Briefly, we extend the original one-dimensional network structure to two-dimensional one and set random connections from model Golgi cells (GOCs) to GRCs (Fig. 1). We also extend their single-compartment GOC model to a multi-compartment model composed of a soma and a dendrite, on which voltage-gated *N*-methyl-*D*-aspartate (NMDA) channels are distributed. We change values of some model parameters so that the network behaves stably.

2.1 Network Structure

In the present model, 32×32 model GOCs are arranged in two-dimensional grids (Fig. 1), in which the model GOCs are evenly positioned at 35 μm intervals within the square sheet of 1,085×1,085 μm^2. It was estimated that there were 1000 times more GRCs than GOCs [8]. Numerous GRCs were connected with a glomerulus [9]. However, simulation with more than 1 million model neurons is beyond the power of our computers. In Yamazaki and Tanaka [10], 100 nearby GRCs that were assumed to be connected with a glomerulus exhibited similar firing patterns despite each of the GRCs received noisy signals through MFs independently. Such redundant activity patterns of GRCs suggest that many GRCs behave as a single cluster when they receive inputs from a nearest GOC through a single glomerulus. In the present model, for the sake of the economy of computer power, we assume that a single model GRC represents a GRC cluster composed of about 1,000 neurons (Fig. 1).

We assume that a GOC receives 9×32 PF inputs from model GRCs with its dendritic arborization whose diameter is set at 315 μm, and the connection probability

of a PF at the GOC is set at 0.1. The model GOC, in turn, sends inhibitory inputs to model GRCs located within the extent of axonal arborization, which is set at 315 μm, so that a model GRC receives inhibitory inputs from 69 nearby model GOCs (Fig. 1). The connection probability from a GOCs to a model GRCs is set at 0.1.

The model network contains 1 PC and 1 SC (Fig. 2). The model PC and SC receive PF inputs from GRCs and climbing fiber inputs that convey US signals. The SC sends an inhibitory connection to the PC.

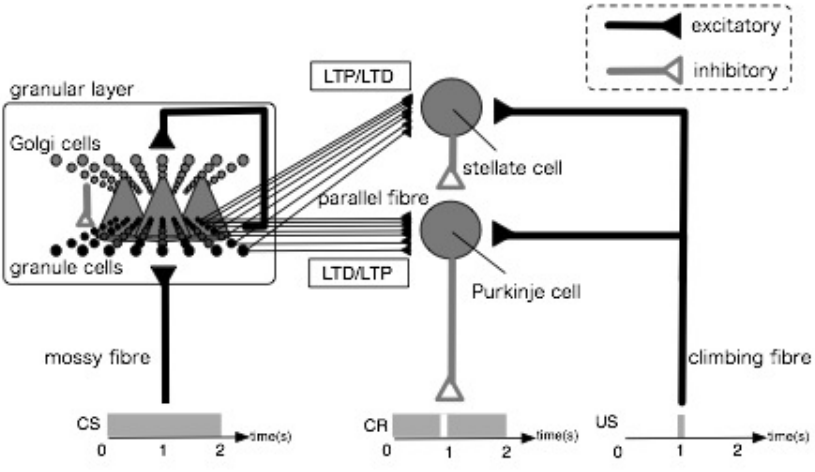


Fig. 1. A schematic of the cerebellar network model. GOCs receive excitatory inputs from GRCs and recurrently inhibit GRCs. Thus, GRCs and GOCs construct a recurrent inhibitory network. Grcs receive CS signals and, in turn, excite a PC and an SC. Paring of US signals fed to a PC and SC with CS signals induce LTD and LTP at PF terminals at these cells.

2.2 Granule and Golgi Cell Models

We use a model GRC composed of a single-compartment Hodgkin-Huxley unit, as adopted by Maex and De Schutter [7]. We simulate inhibitory postsynaptic potential (IPSP) induced by γ-aminobutyric acid type A ($GABA_A$) receptor/channels using a double-exponential function with rise and decay time constants of 5 ms and 100 ms, respectively.

On the other hand, a GOC is modeled as a multi-compartment Hodgkin-Huxley unit composed of a soma and a dendrite, rather than a single-compartment unit as in Maex and De Schutter [7]. The model GOCs receive excitatory inputs from model GRCs through α-amino-3-hydroxy-5-methyl-4-isoxazolepropionic acid (AMPA) channels with rise and decay time constants of 0.03 ms and 0.5 ms, respectively.

It has been found that GOCs receive excitatory input signals from PFs through not only AMPA but also NMDA channels [11]. However, even when we add NMDA channels at the somas of model GOCs, the NMDA channels do not open effectively to evoke sustained depolarization because after-hyperpolarization (AHP) [12][13] following each action potential generation rapidly decreases somatic membrane

potential. On the other hand, it is known that the dendritic potential tends not to be affected by AHP at cell somas [12]. Moreover, it has been shown that direct dendritic excitation produces sustained and burst responses although somatic excitation does not [12]. If NMDA channels really function to induce a prolonged activation of GOCs, it implies that NMDA channels are located on the GOC dendrites. In this study, a model GOC is represented as a soma and a dendrite whose length is 300 μm (cf. [1]) (Fig. 2). The dendrites of model GOCs are assumed to possess AMPA and NMDA channels. We simulate NMDAR-mediated EPSPs with a double-exponential function with rise and decay time constants of 5 ms and 100 ms, respectively, according to Misra et al. [14].

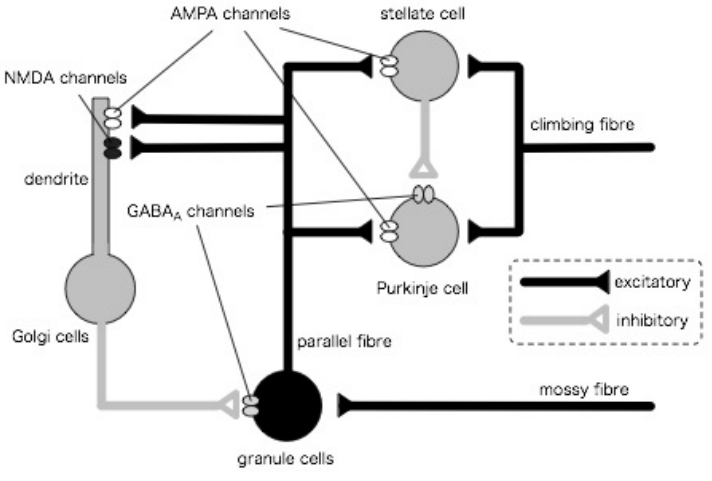


Fig. 2. Grcs' excitation of GOCs is mediated by AMPA and NMDA channels, whereas GOCs' inhibition of GRCs is mediated by $GABA_A$ channels. The strength of current into GRCs, which is induced by MF signal, is external stimulus strength as a control parameter in this model. A PC and an SC are excited by the GRCs mediated by AMPA channels. The PC is inhibited by the SC mediated by $GABA_A$ channels.

2.3 Purkinje and Stellate Cell Models

A PC and an SC are modeled as single-compartment Hodgkin-Huxley units. These cells are assumed to be excited by PF inputs from GRCs mediated by the activation of AMPA channels. The PC is assumed to be inhibited by the SC through $GABA_A$ channels.

2.4 Modeling of a CS

We model MF input signals as current injected directly to the model GRCs, instead of spike trains. Freeman and Muckler [15] have reported that the spontaneous firing rate of MFs is as low as 5 Hz, whereas the firing rate increases up to 30 Hz when stimulated with a tone (e.g. CS). We assume that the current into GRCs increases with the

frequency of firing conveyed through MFs. In simulations, we inject a current I_{MF} of 10.7 pA to all model GRCs for 2 s to simulate GRCs' activities induced by the spontaneous MF activity. For the succeeding 2 s, we inject a current I_{MF} of 29.5 pA to evoke GRCs in response to high frequency MF firing induced by the presentation of the CS. Subsequently, we inject a current I_{MF} of 10.7 pA for 2 s again to resume the baseline activity of GRCs induced by spontaneous MF activity. After conditioning, to simulate network dynamics under CSs of different strengths, we inject current of 31.0 pA or 28.5 pA into GRCs.

2.5 Simulation of Eyeblink Conditioning

For eyeblink conditioning, we present US signal 650 ms after the onset of CS signal presentation. The US signal is sent to both PC and SC. Before conditioning, we set synaptic weights of all PFs at $w_i^{(0)}=1$ and $v_i^{(0)}=1$ for the PC and the SC, respectively. During conditioning, conjunctive stimulation of a CS and a US is assumed to induce LTD and LTP at PF synapses to the PC and the SC, respectively. According to Equations (1) and (2), these synaptic weights are decreased at the PC and increased at the SC when the PFs are activated 50-100 ms before the onset of a US. In addition, we also assume that when spike activity is transmitted to the PC and the SC through PFs alone, PF synapses undergo LTP and LTD, respectively.

$$w_i^{(Tr)} = w_i^{(Tr-1)} 0.9^{f} 1.0005^{f_{tatal}-f}. \tag{1}$$

$$v_i^{(Tr)} = v_i^{(Tr-1)} 1.07^{f} 0.985^{f_{total}-f}. \tag{2}$$

Here, f is the frequency of firing of ith GRC in the interval of 50-100 ms immediately before the onset of a US. f_{tatal} is the total frequency of firing during one trial of conditioning for 2 s. Tr is the trial number. For simplicity, we assume that the US through the climbing fibers affects only learning of PF synaptic weights but not the neurons' activity.

2.6 Data Analysis

When ith GRC $(1 \le i \le N)$ elicits spike activity at time t in response to injected current I^{MF}, $f_i^{(I^{MF})}(t)=1$, and otherwise, $f_i^{(I^{MF})}(t)=0$. We define similarity index (SI) as follows:

$$SI(t_1,t_2) = \frac{\sum_{i=1}^{N} f_i^{I_1^{MF}}(t_1) f_i^{I_2^{MF}}(t_2)}{\sqrt{\sum_{i=1}^{N}\left[f_i^{I_1^{MF}}(t_1)\right]^2}\sqrt{\sum_{i=1}^{N}\left[f_i^{I_2^{MF}}(t_2)\right]^2}}, \tag{3}$$

which represents correlation of a spiking GRC population at time t_1 for CS current I_1^{MF} and that at time t_2 for CS current I_2^{MF}. $SI(t_1,t_2)=1$ indcates that active GRC populations at t_1 and t_2 are identical, where $SI(t_1,t_2)=0$ indicates that completely different GRCs elicit spikes at t_1 and t_2.

3 Results

3.1 POT-Representation

Figure 3*A* represents spike patterns of 20 model GRCs in response to the sustained injection of current of 3 different strengths for simulated CSs. Activation of GRCs by the CS presentation vigorously depolarizes randomly connected GOCs, resulting in the activation of voltage-gated NMDA channels on their dendrites. Because of the long decay time constant of NMDAR-mediated EPSPs, GOCs inhibit nearby GRCs, so that GRCs exhibited random alternations between burst and silent states, as reported by Yamazaki and Tanaka [10].

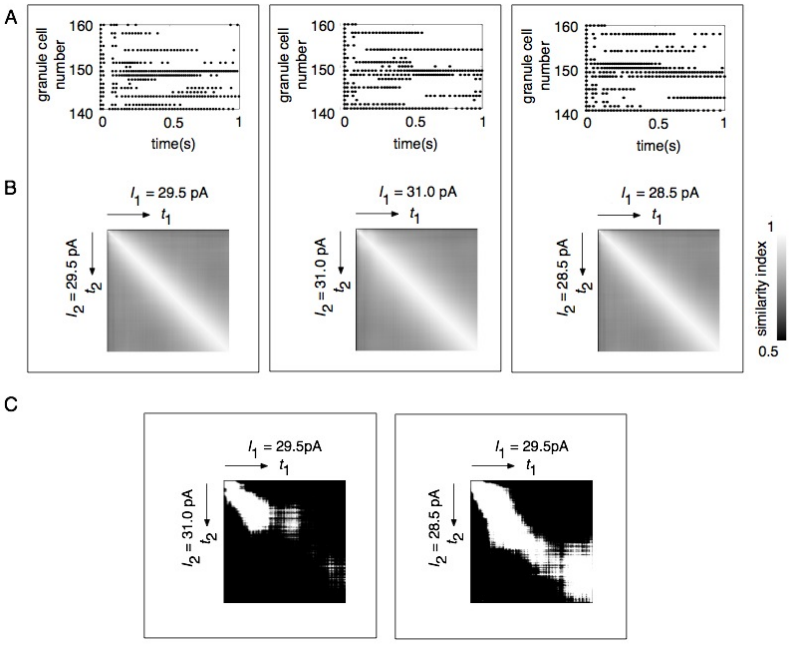


Fig. 3. Network dynamics when current of different strengths is injected to GRCs. (*A*) Spike patterns of 20 GRCs. The abscissa and ordinate represent time and neuron index, respectively. (*B*) *SIs* of spike patterns of GRCs for the same strength of current are plotted in a gray scale between 0.5 and 1. (*C*) *SIs* of spike patterns of GRCs evoked between different strengths of current are plotted in a binary scale in which black (white) indicates $SI \le 0.8$ ($SI > 0.8$).

We show *SIs* of spike patterns of GRCs calculated with Equation 3 in a gray scale (Fig. 3*B*) between 0.5 and 1. Even if the strength of injected current was different, larger *SI* appeared along the diagonal line and gradually decreased with the separation from the diagonal line. This shows that different GRC populations were activated at different times, so that the population of active GRCs changed gradually with time from the CS onset without recurrence. Therefore, the absence of recurrent populations

of active GRCs indicates one-to-one correspondence between a certain population of active GRCs and a certain time, while the sustained current was injected as a CS signal. Thereby, the sequence of active GRC populations is able to represent POT from the CS onset.

3.2 Speeding-Up and Slowing-Down of an Internal Clock

Next, we compared a sequence of GRC populations activated by the sustained injection of current of 29.5 pA with another sequence generated by the injection of current of 31.0pA (28.5 pA). Figure 3*C* shows corresponding SI diagrams. The white band representing high similarity slanted slightly above (below) the diagonal line for the injection of stronger (weaker) current. This indicates that when strong (weak) current is injected, the sequence is compressed (extended). Namely, when input current is stronger (weaker), active GRC populations appear earlier (later) and the speed of generation of active GRC populations becomes faster (slower).

3.3 Simulation of Delay Eyeblink Conditioning

We conducted simulations of delay eyeblink conditioning, in which current injection representing a CS generates a non-recurrent sequence of active GRC populations and a US presented after the CS onset changes synaptic weights of PFs from model GRCs to the model PC and the SC inhibiting the PC.

Before conditioning, the model PC fired at high frequency during the CS presentation (injected current: 29.5 pA). In conditioning, a US signal was given 650 ms after the CS onset. For the model PC, LTD was induced at PF synapses connecting from GRCs active at the timing of the US presentation. For the SC, LTP was induced at PF synapses from those active GRCs.

After 50 trials of conditioning, when we injected current of the same strength as in the conditioning as a CS signal, the PC began to stop firing between 516 ms and 754 ms from the CS onset (Fig. 4*A*). This result indicates that the model PC was able to learn an ISI between CS and US onsets, as consistent with an experiment [16]. On the other hand, when we injected current of 31.0 pA (28.5 pA) as a strong (weak) CS signal, the PC began to stop firing 161ms earlier (898 ms later) than in the default (Fig. 4*B* and *C*). These observations indicate that after conditioning, when a stronger (weaker) CS signal is presented than a CS signal used in conditioning, a PC stopped firing earlier (later). This suggests that even after conditioning, the timing when a CR is elicited can be controlled by the strength of a CS signal.

Next, we examined how the model SC is involved in this timing control after conditioning, by removing the SC after conditioning. First, we injected current of 29.5 pA, the PC began to stop firing between 541 ms and 652 ms after the CS onset (Fig. 4*D*). Although the duration of the PC firing cessation was shorter than that in the default case, the qualitative features of the firing was preserved without the SC. However, when we injected current of 31.0 pA or 28.5 pA after conditioning, the PC kept tonic firing, indicating that the ISI coding was disrupted (Fig. 4*E*, *F*). This suggests that the SC plays an important role for adaptive timing control. Also, when we blocked the plasticity of PF synapses at the SC during conditioning, the PC behaved similarly to the case which a model SC was removed (data not shown).

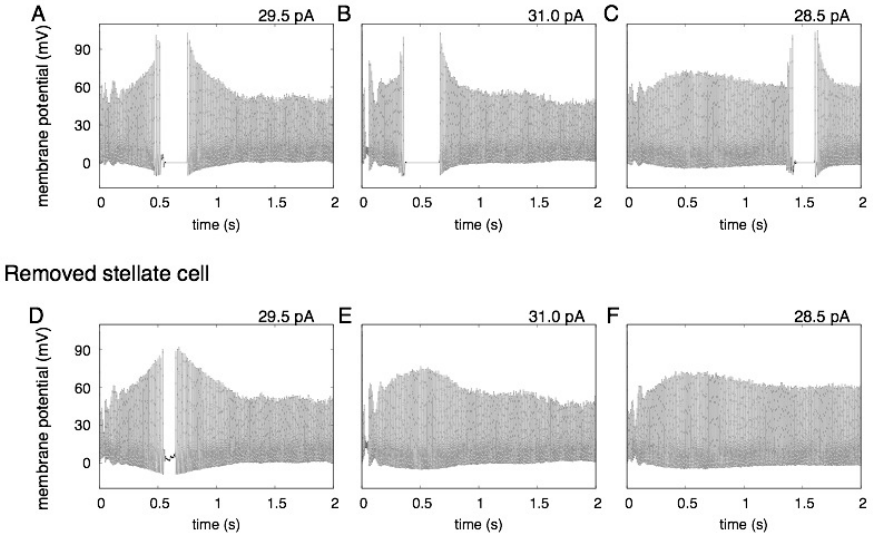


Fig. 4. Membrane potential of the model PC after eyeblink conditioning. In conditioning, US signal is fed to the PC at 650 ms after the onset of sustained injection of current of 29.5 pA to GRCs. (*A*) Membrane potential in response to the current of 29.5 pA after conditioning and (*D*) membrane potential when we remove the SC. (*B*) Membrane potential in response to sustained injection of current of 31.0 pA and (*E*) membrane potential when we remove the SC. (*C*) Membrane potential in response to the current of 28.5 pA and (*F*) membrane potential when we remove the SC. The abscissa represents time from the onset of current injection, and the ordinate represents membrane potential.

4 Discussion and Conclusion

Adaptive control of timing by the strength of external stimulus seems essential for the generalization of motor actions. For example, in batting, we swing a bat at an appropriate timing to hit the ball, estimating the speed of a pitched ball visually. If the speed of a pitched ball is high, we start swinging the bat faster unconsciously. It is expected that the cerebellum learns the timing of motor actions in a supervised manner and controls the learned timing adaptively [17]. In the present study, using our spiking network model of the cerebellar cortex, we argued that a stronger or weaker CS signal conveyed by MFs makes the POT representation change by compressing or expanding the temporal sequence of active GRC populations. Learned timing was advanced or delayed as shown by the temporal shift of timed pause of the PC in the simulated Pavlovian delay eyeblink conditioning, as consistent with experimental findings [18]. We are particularly interested in the role of SCs in the adaptive control of the learned timing. We demonstrated that removing the model SC from the network disrupts the expression of the timing, suggesting that SCs assist the coding of a timing by PCs in motor control, as hypothesized by Rancillac and Crépel [19]. To date, little attention has been paid to the functional role of SCs, except a classical hypothesis as a gain controller of PCs to keep the activity level of PCs within

a physiological range by inhibiting them in a feed-forward manner [20] [21]. The present study shed light for the first time on the functional role of SCs in cerebellar adaptive timing control.

The cerebellum is regarded as a universal simulator that simulates the dynamics of physical and mental objects by acquring the internal models through supervised learning [22] [23]. The present network model of the cerebellar cortex consists of the granular layer that generates a temporal sequence of active GRC populations as liquid states, and a unit composed of a PC and an SC that receives the sequence and extracts time-varying information from the sequence. Once we interpret the granular layer and a PC-SC unit respectively as a reservoir and a readout, our model turns out to be equivalent to a liquid state machine (LSM) [24], a type of artificial neural network model, which is a universal supervised learning machine [25]. In conventional LSMs, a readout is simply a single model neuron that recruits a simple perceptron learning scheme. In contrast, the readout in our model is a pair of two neurons recruiting plasticity of the opposite direction. By this combination, the model cerebellar cortex is able to generalize the timing of a CR adaptively by the strength of a CS signal. This suggests the enhancement of the ability of reading out information from the liquid state in the granular layer. Therefore, the present study may provide insight into the computational power of the cerebellum as a biological counterpart of a liquid state machine. This study also suggests that LSMs can be improved by incorporating inhibitory interneurons in the readout unit.

References

1. Ito, M.: Cerebellum and Neural Control. Raven Press, Hewlett (1984)
2. Yamazaki, T., Tanaka, S.: Neural Modeling of an Internal Clock. Neural Comput. 17, 1032–1058 (2005)
3. Mauk, M.D., Donegan, N.H.: A Model of Pavlovian Eyelid Conditioning Based on the Synaptic Organization of the Cerebellum. Learn. Mem. 3, 130–158 (1997)
4. Hesslow, G., Yeo, C.H.: The Functional Anatomy of Skeletal Conditioning. In: A Neuroscientist's Guide to Classical Conditioning. Springer, New York (2002)
5. Christian, K.M., Thompson, R.F.: Neural Substrates of Eyeblink Conditioning: Acquisition and Retention. Learn. Mem. 11, 427–455 (2003)
6. De Zeeuw, C.I., Yeo, C.H.: Time and Tide in Cerebellar Memory Formation. Curr. Opin. Neurobiol. 15, 667–674 (2005)
7. Maex, R., De Schutter, E.: Synchronization of Golgi and Granule Cell Firing in a Detailed Network Model of the Cerebellar Granule Cell Layer. J. Neurophysiol. 80, 2521–2537 (1998)
8. Palkovits, M., Magyar, P., Szentágothai, J.: Quantitative Histological Analysis of the Cerebellar Cortex in the Cat II Cell Numbers and Densities in the Granular Layer. Brain Res. 32, 13–32 (1971)
9. Mugnaini, E., Atluri, R.L., Houk, J.C.: Fine Structure of Granular Layer in Turtle Cerebellum with Emphasis on Large Glomeruli. J. Neurophysiol. 37, 1–29 (1974)
10. Yamazaki, T., Tanaka, S.: A Spiking Network Model for Passage-of-time Representation in the Cerebellum. Eur. J. Neurosci. 26, 2279–2292 (2007b)
11. Dieudonné, S.: Submillisecond Kinetics and Low Efficacy of Parallel Fibre-Golgi Cell Synaptic Currents in the Rat Cerebellum. J. Physiol. 510, 845–866 (1998)

12. Wong, R.K.S., Stewart, M.: Different Firing Patterns Generated in Dendrites and Somata of CA1 Pyramidal Neurones in Guinea-pig Hippocampus. J. Physiol. 457, 675–687 (1992)
13. Stuart, G.J., Sakmann, B.: Active Propagation of Somatic Action Potentials into Neocortical Pyramidal Cell Dendrites. Nature 367, 69–72 (1994)
14. Misra, C., Brickley, S.G., Farrant, M., Cull-Candy, S.G.: Identification of Subunits Contributing to Synaptic and Extrasynaptic NMDA Receptors in Golgi Cells of the Rat Cerebellum. J. Physiol. 1, 147–162 (2000)
15. Freeman, J.H., Muckler Jr., A.S.: Developmental Changes in Eyeblink Conditioning and Neuronal Activity in the Pontine Nuclei. Learn. Mem. 10, 337–345 (2003)
16. Jirenhed, D., Bengtsson, F., Hesslow, G.: Acquisition, Extinction, and Reacquisition of a Cerebellar Cortical Memory Trace. J. Neurosci. 27, 2493–2502 (2007)
17. Ivry, R.B., Spencer, R.M.: The Neural Representation of Time. Curr. Opin. Neurobiol. 14(2), 225–232 (2004)
18. Svensson, P., Ivarsson, M., Hesslow, G.: Involvement of the Cerebellum in a New Temporal Property of the Conditioned Eyeblink Response. Prog. Brain Res. 124, 317–323 (2000)
19. Rancillac, A., Crépel, F.: Synapses between Parallel Fibres and Dtellate Cells Express Long-term Changes in Synaptic Efficacy in Rat Cerebellum. J. Physiol. 554(Pt. 3), 707–720 (2004)
20. Marr, D.: A Theory of Cerebellar Cortex. J. Physiol. (Lond) 202, 437–470 (1969)
21. Albus, J.S.: A Theory of Cerebellar Function. Math. Biosci. 10, 25–61 (1971)
22. Daniel, M., Wolpert, R., Miall, C., Kawato, M.: Internal Models in the Cerebellum. Trends in Cognitive Sciences 2(9), 338–347 (1998)
23. Ito, M.: Control of Mental Activities by Internal Models in the Cerebellum. Nat. Rev. Neurosci. 9(4), 304–313 (2008)
24. Yamazaki, T., Tanaka, S.: The Cerebellum as a Liquid State Machine. Neural Networks 20(3), 290–297 (2007a)
25. Maass, W., Natschläger, T., Markram, H.: Real-time Computing without Stable States: a New Framework for Neural Computation Based on Perturbations. Neural Comput. 14(11), 2531–2560 (2002)

Parametric Sensitivity and Scalability of k-Winners-Take-All Networks with a Single State Variable and Infinity-Gain Activation Functions*

Jun Wang[1,2] and Zhishan Guo[1]

[1] Department of Mechanical and Automation Engineering,
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[2] Department of Computer Science and Engineering,
Shanghai Jiao Tong University, Minhang, Shanghai, China
{jwang,zsguo}@mae.cuhk.edu.hk

Abstract. In recent years, several k-winners-take-all (kWTA) neural networks were developed based on a quadratic programming formulation. In particular, a continuous-time kWTA network with a single state variable and its discrete-time counterpart were developed recently. These kWTA networks have proven properties of global convergence and simple architectures. Starting with problem formulations, this paper reviews related existing kWTA networks and extends the existing kWTA networks with piecewise linear activation functions to the ones with high-gain activation functions. The paper then presents experimental results of the continuous-time and discrete-time kWTA networks with infinity-gain activation functions. The results show that the kWTA networks are parametrically robust and dimensionally scalable in terms of problem size and convergence rate.

Keywords: K winners-take-all, recurrent neural networks, optimization, scalability, parametric sensitivity.

1 Introduction

As a generalization of the winner-take-all (WTA) operation [1], the k-winners-take-all (kWTA) operation selects the k largest inputs out of n inputs ($1 \leq k \leq n$). kWTA has been shown to be a computationally powerful operation compared with standard neural network models of threshold logic gates [2]. In addition, it has numerous applications in k-neighborhood classification, k-means clustering, sorting, machine learning, and data mining; e.g., decoding [1], image processing [3], computer vision [4],[5], feature extraction [6], and associative memories [7], mobile robot navigation [8], etc.

* The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project no. CUHK417608E).

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 77–85, 2010.

When the number of inputs is large or a selection process has to be operated in real time, parallel algorithms and hardware implementation are desirable. In the past twenty years, many WTA and kWTA networks have been developed; e.g., [9]-[25]. In particular, a continuous-time kWTA network with a single state variable and guaranteed global convergence [23] was developed based on a quadratic programming formulation in [20]. It has the simplest architectural complexity due to its single state variable. A discrete-time counterpart was developed recently [25].

In this paper, we will present the simulation results of these kWTA networks to show their parametric robustness and dimension scalability in terms of convergence, which will validate and supplement previous theoretical results for designing the kWTA networks.

2 Problem Formulations

Generally, the kWTA operation can be defined or encoded as the following binary function

$$x_i = f(u_i) = \begin{cases} 1, \text{ if } u_i \in \{k \text{ largest elements of } u\}, \\ 0, \text{ otherwise}, \end{cases} \tag{1}$$

where $u = (u_1, u_2, \ldots, u_n)^T$ is the input vector and $x = (x_1, x_2, \ldots, x_n)^T$ is the output vector.

The kWTA solution can be determined by solving the following linear integer programming problem:

$$\begin{aligned} &\text{maximize } \sum_{i=1}^{n} u_i x_i \quad \text{or} \quad \text{minimize} - \sum_{i=1}^{n} u_i x_i, \\ &\text{subject to } \sum_{i=1}^{n} x_i = k, \\ &\qquad\qquad x_i \in \{0, 1\}, \quad i = 1, 2, \ldots, n. \end{aligned} \tag{2}$$

According to its total modularity property [26], the above linear integer programming problem is equivalent to the following linear programming problems if the kth and $(k+1)$th largest elements of u are different (i.e., the solution is unique):

$$\begin{aligned} &\text{minimize } -\sum_{i=1}^{n} u_i x_i \quad \text{or} \quad -u^T x, \\ &\text{subject to } \sum_{i=1}^{n} x_i = k, \\ &\qquad\qquad 0 \le x_i \le 1, \quad i = 1, 2, \ldots, n. \end{aligned} \tag{3}$$

It was proven in [20] that the kWTA problem is equivalent to the following quadratic integer programming problems:

$$\begin{aligned} &\text{minimize } \tfrac{a}{2} \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} u_i x_i \text{ or } \tfrac{a}{2} x^T x - u^T x, \\ &\text{subject to } \sum_{i=1}^{n} x_i = k, \\ &\qquad\qquad x_i \in \{0, 1\}, \quad i = 1, 2, \ldots, n. \end{aligned} \tag{4}$$

where a is a positive constant.

Furthermore, according to [20], if the kth and $(k+1)$th largest elements of u (denoted as $\bar{u}_k$ and $\bar{u}_{k+1}$, respectively) are different, the quadratic integer programming problems is equivalent to the following quadratic programming problems:

$$\begin{aligned} &\text{minimize} \quad \tfrac{a}{2}\sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} u_i x_i \text{ or } \tfrac{a}{2}x^T x - u^T x, \\ &\text{subject to} \sum_{i=1}^{n} x_i = k, \\ &\qquad\qquad 0 \le x_i \le 1, \quad i = 1, 2, \ldots, n, \end{aligned} \tag{5}$$

where $a \le \bar{u}_k - \bar{u}_{k+1}$ represents the input resolution.

3 Model Descriptions

A kWTA neural network was tailored from the simplified dual network [20]:

$$\text{state equation} \quad \epsilon\frac{dv}{dt} = -Mv + g(Mv - v + s) - s, \tag{6}$$

$$\text{output equation} \quad x = Mv + s, \tag{7}$$

where $\epsilon > 0$ is a scaling constant, $v \in \Re^n$ is the state vector, $M = (I_n - ee^T/n)/a$, $s = Mu + (k/n)e$, e is a column vector with all entries being ones, and $g(\cdot)$ is the piecewise linear activation function defined as:

$$g(x_i) = \begin{cases} 0, & x_i < 0_i, \\ x_i, & 0 \le x_i \le 1, \\ 1, & x_i > 1. \end{cases} \tag{8}$$

A kWTA network with a single state variable was developed based on the improved dual neural network [23] with the following following equations:

$$\text{state equation} \quad \epsilon\frac{dz}{dt} = -\sum_{i=1}^{n} x_i + k, \tag{9}$$

$$\text{output equation} \quad x_i = g\left(z + \frac{v_i}{a}\right), \quad i = 1, \cdots, n, \tag{10}$$

where $z \in \Re$ is the state variable. The exactly same model was reinvented one year after in [24].

Recently, a discrete-time counterpart of the above kWTA network [23] was presented [25],[24]..

$$\text{state equation} \quad z(m+1) = z(m) - \beta(\sum_{i=1}^{n} x_i(m) - k), \tag{11}$$

$$\text{output equation} \quad x_i(m) = g\left(z(m) + \frac{v_i}{a}\right), \quad i = 1, \cdots, n, \tag{12}$$

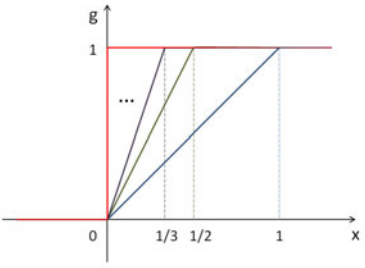


Fig. 1. The piecewise linear activation function with a positive gain parameter ξ

where $\beta > 0$ is the step size. It was proven that the discrete-time kWTA network is globally convergent if $0 < \beta < 2/n$ [25] or $0 < \beta < 2/\sqrt{n}$ [24].

As the output variables are supposed to be binary, the piecewise linear activation function $g(\cdot)$ can be extended with a positive gain parameter ξ as shown in(13) and Figure 1, where ξ is the slope of the linear part:

$$g(x_i) = \begin{cases} 0, x_i < 0_i, \\ \xi, 0 \leq x_i \leq 1/\xi, \\ 1, x_i > 1/\xi. \end{cases} \tag{13}$$

When the gain parameter approaches to positive infinity (i.e., $\xi \to +\infty$), the activation function becomes identical with that in the one of one-layer neural networks with hard-limiting (step) activation function in [22]. In the next section, it will be shown that the kWTA networks with positive infinity gain parameter ξ can still work well in kWTA operations.

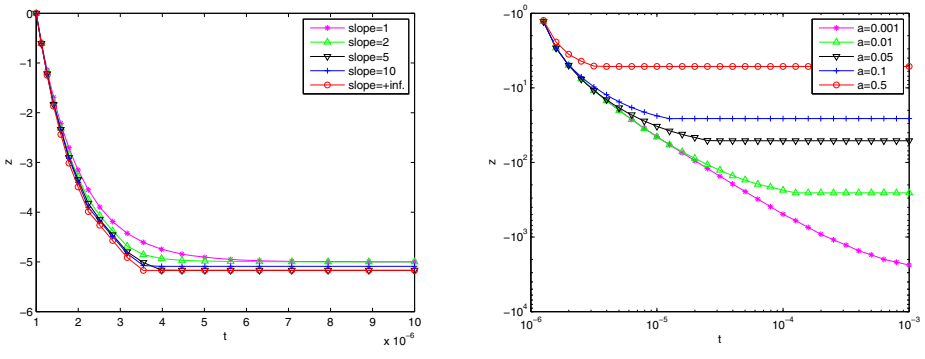


Fig. 2. Stable state variable $z(t)$ with respect to various different values of parameters ξ and a in the continuous-time kWTA network (9)

4 Simulation Results

In this section, the results of extensive simulations will be discussed with respect to various parameters of the kTWA networks (e.g., ξ, a, ϵ, $z(0)$, and n). In all simulations, unless otherwise specified, the inputs are set as $u_i \in \{1, 2, \ldots, n\}$ and the parameters are set as $n = 10, k = 5, a = 0.5, z(0) = 0, \epsilon = 10^{-6}, \xi \to +\infty$.

In Fig. 2, the transient behaviors of the state variable $z(t)$ of the continuous-time kWTA network in (9) are depicted with respect to increasing gain parameter ξ from 1 to $+\infty$ in the left subplot and various values of parameter a ranging from 0.001 to 0.5 in the right subplot where both axes are in a logarithmic scale. The results show that the convergence is faster when the gain parameter ξ is larger or a is smaller. In Fig. 3, the transient behaviors of the state variable $z(t)$ of the continuous-time kWTA network in (9) are depicted with respect to 30

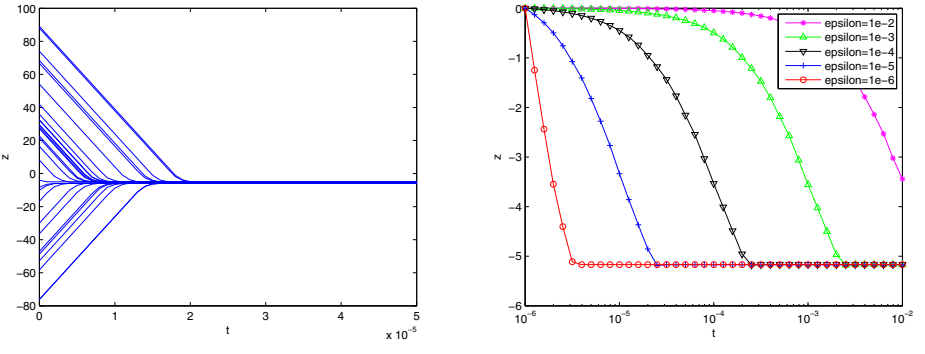


Fig. 3. Stable state variable $z(t)$ with respect to various different values of initial state $z(0)$ and scaling constant ϵ in the continuous-time kWTA network (9)

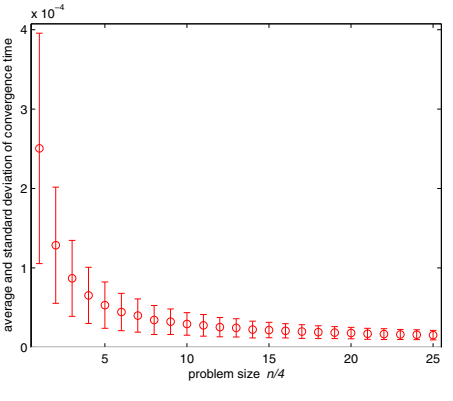


Fig. 4. Average and deviation of convergence time in the continuous-time kWTA network (9) with 1000 random inputs u of increasing size n, where $k = n/2$

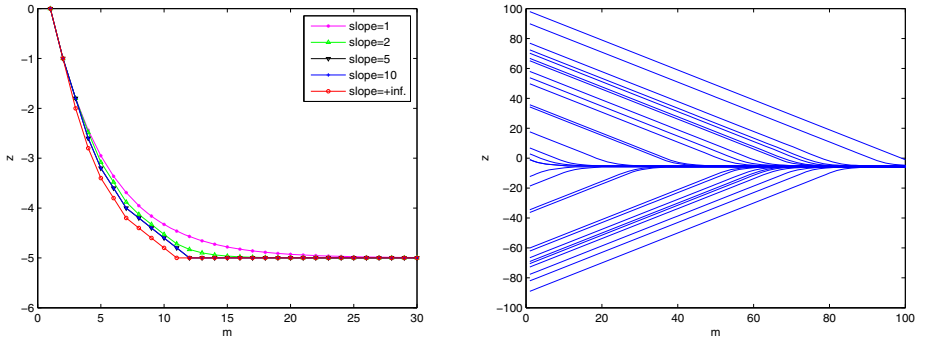


Fig. 5. Stable state variable $z(m)$ with respect to various different values of gain parameter ξ and initial state $z(0)$ in the discrete-time kWTA network (11)

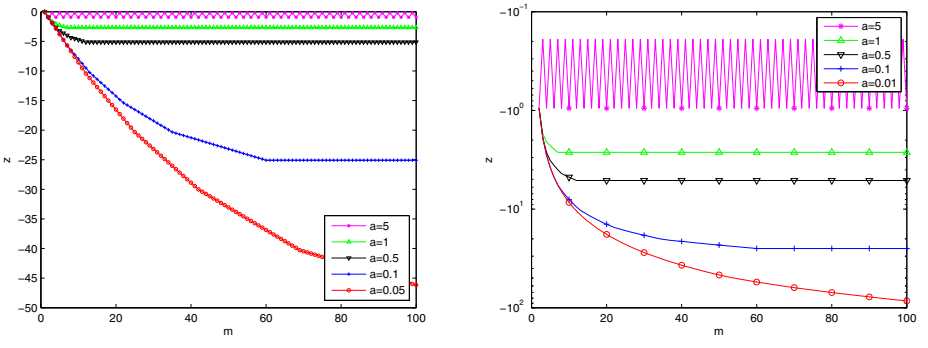


Fig. 6. Stable state variable z with respect to various different values of resolution parameter a in the discrete-time kWTA network (11)

random initial states $z(0)$ drawn from [-100,100] under the uniform distribution in the left subplot and decreasing scaling constant ϵ from 10^{-2} to 10^{-6} in the right subplot where the horizontal axis is in a logarithmic scale. It is obvious that the state variable z is globally stable from any initial state $z(0)$ and its convergence is faster when the scaling constant ϵ is smaller. In Fig. 4, the Monte Carlo simulation results are shown 1000 random inputs $u_i \in \{1, 2, \ldots, n\}$ with increasing sizes (i.e., $n = 4, 8, \ldots, 100$). It is interesting to see that the average and standard deviation of convergence time of the state variable z decrease as the problem size n increases, which reinforces the results in [23]. In particular, the averaged convergence time approaches to a stable value about 15 microseconds for large kWTA problems.

Similar to the results for the continuous-time kWTA network (9) and (10), the simulation results for the discrete-time kWTA network (11) and (12) are shown in Figs. 5-8. Specifically, Fig. 5 depicts the transient behaviors of the state variable $z(m)$ of the discrete-time kWTA network in (11) with respect

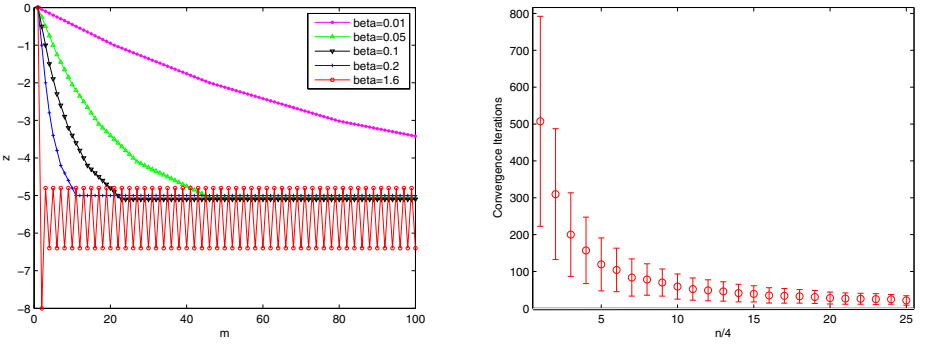


Fig. 7. (i) Stable state variable z with respect to various different values of step size parameter β in the discrete-time kWTA network (11). (ii) Average and deviation of the number of iterations for convergence in the discrete-time kWTA network (11) with 1000 random inputs of increasing size n, where $k = n/2$.

to increasing gain parameter ξ from 1 to $+\infty$ and 30 random initial states $z(0)$ drawn from [-100,100] under the uniform distribution, which show that the convergence is faster when the gain parameter ξ is larger and global stability of the state variable. In Fig. 6, the transient behaviors of state variable $z(m)$ of the discrete-time kWTA network are depicted with respect to various values of parameter a ranging from 0.01 to 5, where the vertical axis for z is in a logarithmic scale in the right subplot. The results show that the convergence of the state variable z is faster when a is larger, which is contrary to that in continuous-time kWTA network shown in Fig. 2 (right subplot). In addition, when a is too big, the state of the discrete-time network becomes unstable. In Fig. 7, the transient behaviors of the state variable $z(m)$ of the discrete-time kWTA network are depicted with respect to various values of its step size parameter β ranging from 0.01 to 1.6. It is not surprising that the convergence is faster when β is bigger and an oscillation occurs when β is too big. In Fig. 8, the Monte Carlo simulation results are shown 1000 random inputs $u_i \in \{1, 2, \ldots, n\}$ with increasing sizes (i.e., $n = 4, 8, \ldots, 100$) for the discrete-time network. It can be observed that, as the problem size n increases, the average number of iterations needed for the of state variable z converges to a stable value 21.50, but its deviation decreases.

5 Concluding Remarks

In this paper, extensive simulation results are reported on the global state stability of the continuous-time and discrete-time k-winners-take-all (kWTA) networks with a single state variable under various parametric configurations. In particular, it is demonstrated that these kWTA networks work faster with high-gain activation functions. In addition, the scalability of the kWTA networks is also shown in terms of input dimension and convergence time. The theoretical

justifications of the high-gain activation function in the kWTA networks will be published in a separate paper.

References

1. Majani, E., Erlanson, R., Abu-Mostafa, Y.: On the k-winners-take-all network. In: Touretzky, D.S. (ed.) Advances in Neural Information Processing Systems, vol. 1, pp. 634–642. Morgan-Kaufmann, San Mateo (1989)
2. Maass, W.: On the computational power of winner-take-all. Neural Comput. 12, 2519–2535 (2000)
3. Fish, A., Akselrod, D., Yadid-Pecht, O.: High precision image centroid computation via an adaptive k-winner-take-all circuit in conjunction with a dynamic element matching algorithm for star tracking applications. Analog Integrated Circuits and Signal Processing 39, 251–266 (2004)
4. Marr, D., Poggio, T.: Cooperative computation of stereo disparity. Science 194(4262), 283–287 (1976)
5. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Trans. Pattern Analysis and Machine Intelligence 20, 1254–1259 (1998)
6. Yuille, A., Geiger, D.: Winner-take-all networks. In: The Handbook of Brain Theory and Neural Networks, 2nd edn., pp. 1228–1231. MIT Press, Cambridge (2003)
7. Pouliquen, P.O., Andreou, A.G., Strohbehn, K.: Winner-takes-all associative memory: a hamming distance vector quantizer. Analog Integrated Circuits and Signal Processing 13, 211–222 (1997)
8. DeSouza, G.N., Kak, A.C.: Vision for mobile robot navigation: a survey. IEEE Trans. Pattern Analysis and Machine Intelligence 24, 237–267 (2002)
9. Wolfe, W., Mathis, D., Anderson, C., Rothman, J., Gottler, M., Brady, G., Walker, R., Duane, G., Algghband, G.: K-winner networks. IEEE Trans. Neural Networks 2, 310–315 (1991)
10. Dempsey, G.L., McVey, E.S.: Circuit implementation of a peak detector neural network. IEEE Trans. Circuits Syst. II 40, 585–591 (1993)
11. Wang, J.: Analogue winner-take-all neural networks for determining maximum and minimum signals. Int. J. Electron. 77, 355–367 (1994)
12. Urahama, K., Nagao, T.: K-winners-take-all circuit with o(n) complexity. IEEE Trans. Neural Networks 6(3), 776–778 (1995)
13. Wu, J.L., Tseng, Y.H.: On a constant-time, low-complexity winner-take-all neural network. IEEE Trans. Computers 44, 601–604 (1995)
14. Yen, J., Guo, J., Chen, H.: A new k-winners-take-all neural network and its array architecture. IEEE Trans. Neural Networks 9(5), 901–912 (1998)
15. Sekerkiran, B., Cilingiroglu, U.: A CMOS k-winners-take-all circuit with O(N) complexity. IEEE Trans. Circuits and Systems 46, 1–5 (1999)
16. Sum, J.P.F., Leung, C.S., Tam, P.K.S., Young, G.H., Kan, W.K., Chan, L.W.: Analysis for a class of winner-take-all model. IEEE Trans. Neural Netw. 10, 64–71 (1999)
17. Calvert, B.A., Marinov, C.: Another k-winners-take-all analog neural network. IEEE Trans. Neural Netw. 11, 829–838 (2000)
18. Marinov, C., Calvert, B.: Performance analysis for a k-winners-take-all analog neural network: basic theory. IEEE Trans. Neural Networks 14(4), 766–780 (2003)

19. Marinov, C.A., Hopfield, J.J.: Stable computational dynamics for a class of circuits with $O(N)$ interconnections capable of KWTA and rank extractions. IEEE Trans. Circuits Syst. I 52, 949–959 (2005)
20. Liu, S., Wang, J.: A simplified dual neural network for quadratic programming with its KWTA application. IEEE Trans. Neural Netw. 17(6), 1500–1510 (2006)
21. Hu, X., Wang, J.: Design of general projection neural networks for solving monotone linear variational inequalities and linear and quadratic optimization problems. IEEE Tran. Systems, Man and Cybernetics - Part B 37, 1414–1421 (2007)
22. Liu, Q., Wang, J.: Two k-winners-take-all networks with discontinuous activation functions. Neural Networks 21(2-3), 406–413 (2008)
23. Hu, X., Wang, J.: An improved dual neural network for solving a class of quadratic programming problems and its k-winners-take-all application. IEEE Trans. Neural Networks 19, 2022–2031 (2008)
24. Xia, Y., Sun, C.: A novel neural dynamical approach to convex quadratic program and its efficient applications. Neural Networks 22(10), 1463–1470 (2009)
25. Liu, Q., Cao, J., Liang, J.: A discrete-time recurrent neural network with one neuron for k-winners-take-all operation. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5551, pp. 272–278. Springer, Heidelberg (2009)
26. Bazaraa, M.S., Jarvis, J.J., Sherali, H.D.: Linear Programming and Network Flows. Wiley, New York (1990)

Extension of the Generalization Complexity Measure to Real Valued Input Data Sets

Iván Gómez, Leonardo Franco, José M. Jerez, and José L. Subirats

Departamento de Lenguajes y Ciencias de la Computación
E.T.S.I. Informática, Campus de Teatinos, Málaga, Spain
{ivan,lfranco,jja,jlsubirats}@lcc.uma.es

Abstract. This paper studies the extension of the Generalization Complexity (GC) measure to real valued input problems. The GC measure, defined in Boolean space, was proposed as a simple tool to estimate the generalization ability that can be obtained when a data set is learnt by a neural network. Using two different discretization methods, the real valued inputs are transformed into binary values, from which the generalization complexity can be straightforwardly computed. The discretization transformation is carried out both through a very simple method based on equal width intervals (EW) and with a more sophisticated supervised method (the CAIM algorithm) that use much more information about the data. A study of the relationship between data complexity and generalization ability obtained was done together with an analysis of the relationship between best neural architecture size and complexity.

Keywords: Neural network, Architecture size, Real-valued function, CAIM, Discretization algorithms.

1 Introduction

Deciding how many nodes to include in the hidden layer of a feed forward neural network in order to classify a set of patterns or to approximate a given data set is a controversial issue. Using a variety of mathematical methods and approximations, theoretical results that give an indication about how to solve this problem have been obtained [1,3,4,5,6,7,8,9,10], but at the time of the implementation some of them offer not clear help or are very difficult to implement. As a consequence, more practical approaches, with results supported by numerical simulations have been proposed [19,17,18]. Despite all these approaches, still the main tendency at the time of selecting a neural network architecture for a given problem is the trial-and-error approach.

Recently, a new point of view to the architecture selection problem have been tried by Franco and colleagues [14,13,12], who have introduced a new measure for the estimating the complexity of Boolean functions. The measure named Generalization Complexity (GC) tries to estimate from the set of available data for a given problem what it will be the generalization ability expected, as several tests have shown that a high correlation exists between the GC measure and the

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 86–94, 2010.

prediction accuracy obtained when Boolean data is used to train a Feed-forward Neural Network (FFNN). As mentioned, the method to select proper neural architectures based on the GC complexity of the data has been only applied to Boolean input data [12], as the GC measure is defined for Boolean inputs. In this work, through the use of two different discretization (or binarization) methods [28], we compute the GC measure of real-valued input data from a well known benchmark data set. 20 classification problems from the standardized PROBEN1 [2] benchmark collection were used as testing data for analyzing the relationship between the GC measure and the generalization ability obtained when the problems are learnt by FFNN, and also we have analyze the relationship between the size of the optimal-found network and the generalization ability obtained. We have used two very different discretization methods, a very simple and unsupervised one, that discretize the inputs using equal width intervals (EW) [24], and a more complex supervised discretization algorithm, named Class-Attribute Interdependence Maximization (CAIM) based on the maximization of the interdependence between class and attributes [27].

The paper is structured as follows: section 2 contains the details of the methods and data sets used, followed by the results obtained from the extensive numerical simulations that are presented in section 3 to finally discuss the results in section 4.

2 Methods and Benchmark Data Sets

2.1 The GC Measure

The GC measure was introduced by Franco and colleagues [14,13], and was derived from evidence that pairs of bordering examples, those lying closely at both sides of separating hyperplanes that classify different regions, play an important role on the generalization ability obtained in classification problems when neural networks are used as predictive methods [15,16] . The GC measure of a Boolean function f, $\mathcal{C}[f]$ can be simply computed by counting the number of pairs of neighboring examples having opposite outputs located at Hamming distances 1 and 2.

$$\mathcal{C}[f] \;=\; \mathcal{C}_1[f] + \mathcal{C}_2[f], \tag{1}$$

where $\mathcal{C}_i[f]$, $i = 1, 2$ are the two terms of the measure taking into account pairs of examples at a Hamming distance one and two. Explicitly, the first term can be written as:

$$\mathcal{C}_1[f] = \frac{1}{N_{ex} \; * \; N_{neigh}} \sum_{j=1}^{N_{ex}} \left(\sum_{\{l | Hamming(e_j, e_l)=1\}} |f(e_j) - f(e_l)| \right) \tag{2}$$

where the first factor, $\frac{1}{N_{ex} \; * \; N_{neigh}}$, is a normalization one, counting for the total number of pairs considered, N_{ex} is the total number of examples equals to 2^N,

and N_{neigh} stands for the number of neighbor examples at a Hamming distance of 1. The second term $\mathcal{C}_2[f]$ is constructed in an analogous way.

In order to apply the GC measure for the estimation of the size of a proper neural network architecture, a fit of the relationship between architecture and GC has to be obtained for a set of training functions, to then use this fit to estimate an adequate architecture according to the GC complexity of a new data set. This work was carried out in [12] and the results compared to other existing approaches, finding that the GC-based method works close to the optimal.

2.2 Binarization of the Data Set: Equal Width and CAIM Algorithms

As mentioned before, the GC measure has only be defined for binary input data, and thus in order to transform real-valued inputs to binary values a discretization (or binarization) algorithm should be applied.

Usually, discretization algorithms are classified in two main categories: unsupervised and supervised algorithms.

- Unsupervised algorithms discretize attributes without taking into account respective class labels. They are the simplest to use and implement. The most representative algorithms of this category are equal-width and equal-frequency methods. The equal-width discretization algorithm find the minimum and maximum values for each attribute, and then divides this range into a number n_{F_i} of user specified equal width intervals. The equal-frequency discretization algorithm determines the minimum and the maximum values of the attribute, sort all values in ascending order, and divides the range into a user-defined number of intervals so that every interval contains the same numbers of sorted values.
- Supervised algorithms discretize attributes by taking into account the interdependence between class labels and the attribute values. The representative algorithms are: maximum entropy [20], Patterson and Niblett [21], information entropy maximization (IEM) [22], and other information-gain or entropy-based algorithms [23], statistics-based algorithms like ChiMerge [24], and clustering-based algorithms like K-means discretization [25].

  CAIM [27](Class-Attribute Interdependence Maximization) is an algorithm designed to work with supervised data. The goal of the CAIM algorithm is to maximize the class-attribute interdependence and at the same time generate a minimal number of discrete intervals. A main advantage of the CAIM algorithm is that does not require to predefine the number of intervals, as opposed to some other discretization algorithms, as CAIM automatically selects the number of intervals without any user supervision.

In this work, we implemented and applied one unsupervised and one supervised algorithms. The unsupervised algorithm used was an equal width one (EW) that we tried with 3 and 5 intervals. Among the supervised algorithms we choose the CAIM one, as good results have been reported from its performance [27].

2.3 The Benchmark Data Set

In order to investigate the performance of the GC measure over real-valued input functions, we have created several binary classification data sets from real world problems, specifically from the PROBEN1 benchmark data set [2]. This benchmark is a collection of problems for neural network learning in the field of pattern classification and function approximation plus a set of rules and conventions for carrying out benchmark tests. All data sets are represented with real-valued attributes, and they are realistic problems presented in the same simple format. The problem representation in PROBEN1 is one of the best improvements made in the benchmark, being a fixed representation that improves the comparability of results, reducing drastically the work to be done for running the benchmark. The data sets included in the present analysis are: **cancer1, card1, diabetes1, gene1, heart1, heartc1, horse1 and thyroid1**. For simplicity we have considered m-class output data as m different two-class problems and have performed this process for every function in the benchmark for which m was larger than 2. The final data used for the study comprise the 20 two-class classification problems shown in table 1.

The first column in Table 1 shows a problem identifier from 1 to 20, followed by the source function name, the percentage of examples with output class 0 or 1, the number of inputs of the source problem, the number of inputs obtained when the CAIM binarization algorithm is applied and finally the complexity values obtained using the GC measure for the three cases considered. The last three columns show the result of the application of the GC measure to the data after applying the CAIM algorithm and the equal-width method for 3 and 5 intervals. The number of inputs used for the case of the equal width algorithm can be straightforward multiplying by 3 and 5 the source number of inputs, as the EW method creates, for every single input, a number of equivalent inputs equal to the number of intervals considered.

3 Simulations

We carried intensive numerical simulations for the 20 2-class functions described in table 1 analyzing the generalization ability of FFNN as the number of neurons in the single hidden layer is varied between 2 and 30. All the networks were trained with scale conjugate gradient back propagation algorithm [26], due to its modest requirements of memory and the good performance of this algorithm for problems with large number of weights. The maximum number of epochs allowed for the training procedure was set to 5000 and the minimum performance gradient used was $1e-5$, stopping the training if the magnitude of the gradient drops below this value. In order to avoid the problem of overfitting that degrades the generalization ability, we implemented the well known early stopping procedure [11], in which the validation error is monitored during training to then choose the synaptic weights observed at the point where the validation error took its minimum value. A stratified 10-cross validation method was used to train the different networks, with the aim of preserving the percentage of examples with

Table 1. Description of the 20 functions generated from the PROBEN1 benchmark used in the present study. The table shows in the columns a function identifier (id), the original source problem name, the distribution of the output class in 0's and 1's, the source number of inputs, the number of discretized inputs using the CAIM binarization algorithm and in the last three columns the values of the GC measure. The three GC values corresponds to the values obtained from the binarized version of the problem using the CAIM algorithm and the equal width (EW) algorithm for 3 and 5 intervals.

id	**Source**	% **class output (0-1)**	# source inputs	# CAIM inputs	GC-Complexity		
					CAIM	EW-3	EW-5
f_1	cancer1	65 − 35	9	18	0.0525	0.0640	0.0470
f_2	card1	45 − 55	51	100	0.2033	0.2079	0.1969
f_3	diabetes1	35 − 65	8	16	0.3761	0.3911	0.3351
f_4	gene1	76 − 24	120	240	0.1617	0.1617	0.1617
f_5	gene1	76 − 24	120	240	0.1764	0.1764	0.1764
f_6	gene1	48 − 52	120	240	0.2417	0.2417	0.2417
f_7	glass1	67 − 33	9	18	0.3105	0.3449	0.3676
f_8	glass1	64 − 36	9	18	0.4048	0.4100	0.3562
f_9	glass1	92 − 8	9	18	0.1570	0.1440	0.1452
f_{10}	glass1	94 − 6	9	18	0.0591	0.0650	0.0339
f_{11}	glass1	96 − 4	9	18	0.0566	0.0653	0.0651
f_{12}	glass1	86 − 14	9	18	0.0573	0.0919	0.0425
f_{13}	heart1	44 − 56	35	67	0.2412	0.2660	0.2252
f_{14}	heartc1	54 − 46	35	59	0.2333	0.3048	0.2256
f_{15}	horse1	38 − 62	58	116	0.3097	0.3505	0.3075
f_{16}	horse1	75 − 25	58	116	0.2454	0.2582	0.2328
f_{17}	horse1	85 − 15	58	116	0.1861	0.1799	0.1850
f_{18}	thyroid1	97 − 3	21	42	0.0120	0.0449	0.0441
f_{19}	thyroid1	95 − 5	21	42	0.0553	0.0996	0.0993
f_{20}	thyroid1	7 − 93	21	42	0.0543	0.1394	0.1383

different output for each fold. Each 10-fold cross validation was set up choosing iteratively a test fold, then a random validation fold, and the remaining eight folds used for training. The cross validation process was repeated 5 times with a different random seed value, and the final result for each considered network was the mode of these 5 values and then the median of the 10-fold combination. The simulations were run on Matlab code under the Linux operating system in a cluster of 10 blades interconnected with infiniband, each one equipped with 2 Xeon Quadcore processors.

4 Results

In table 1 the values of the GC measure obtained with the different discretization methods used are shown in the last three columns. As it can be appreciated from the table, in almost all cases the complexity values are very similar for the three cases considered. Some differences can be observed for the three last cases analyzed, for which the output class distribution is highly unbalanced.

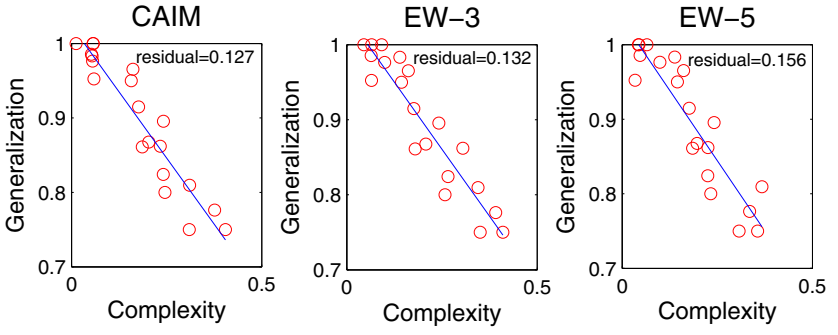


Fig. 1. Generalization ability vs function complexity (GC) for the 20 real valued input functions analyzed for the case of applying the CAIM algorithm and the equal width algorithm with 3 and 5 intervals. The solid curves shown are a linear fit of the results, and the corresponding residual errors are indicated on top of the individual figures.

An analysis of these cases have shown that, for this kind of highly unbalanced output functions, a simple method like the equal interval one, can overestimate the "true" complexity value.

Figure 1 shows the generalization ability obtained as a function of the GC measure for the whole set of test functions using the CAIM and the equal width algorithm with 3 and 5 intervals. A linear fit of the generalization ability as a function of the GC complexity is indicated by a solid line and the residual error obtained indicated on top of each individual graph. The residual error is lower for the CAIM method, indicating a better linear fit between generalization and GC for the case of using the CAIM algorithm. These results confirm the previously ones obtained with true Boolean input functions [14,13,12] and are expected as a lower generalization ability should be obtained for problems with a higher complexity.

Regarding the different discretization methods applied (CAIM and EW with 3 and 5 intervals), we can observe a slightly better fit when the CAIM algorithm is used, but not big differences are observed, especially considering that the CAIM algorithm use much more information about the function than the rather simple EW algorithm.

We have further analyzed the relationship between adequate neural architectures and problem complexity, studying the generalization ability obtained when the problems are implemented in FFNN of varying size. In Figure 2 the values of the number of neurons in the hidden layer of the neural architectures used vs the GC-measure of the data sets are represented. The shown values are for the architectures that produced the best generalization ability when varying the number of hidden neuron in the single hidden layer considered between 2 and 30. In the case of obtaining the same value for the generalization ability, the smaller architecture was chosen following Occam's razor principle: the simpler the solution the better.

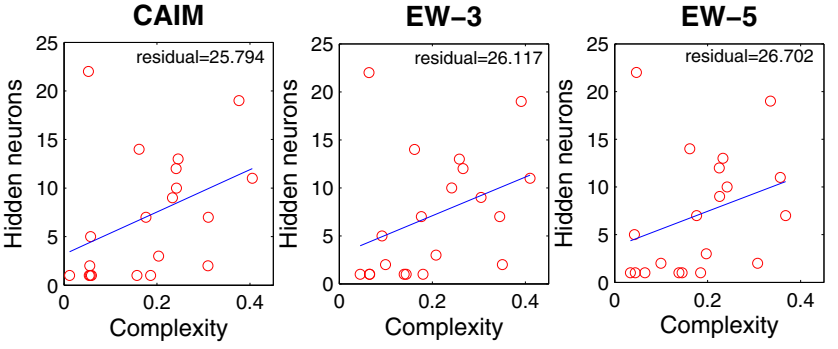


Fig. 2. Best architecture size vs function complexity (GC) for the 20 real world functions described in table 1. The curves are a linear fits of the results obtained when the functions were discretized with the CAIM algorithm, and the 3 and 5 equal-width discretization method.

For the three cases considered and shown in separate graphs in Figure 2, the relationship between the GC measure and the best architecture size was similar, with a slightly better fit found for the case of using the CAIM algorithm as discretization process, as shown by the residual value indicated in the figures. The residual values are somewhat large in all three cases as the fit it is not extremely accurate due to the variability existing in the chosen neural sizes. In particular, note that in each of the three the graphs there is a problem with low GC complexity for which a very large preferred architecture with 22 neurons lead to the best generalization results. This particular value is far apart from the linear fit leading to large residual error. We have observed previously these particular cases, that indicate that the relationship between best architecture and function complexity holds on average but some caution is necessary in individual cases. Nevertheless, we note that in most of these particular cases the difference on generalization between the preferred size network and a linear predicted one is fairly small.

5 Conclusions

Through the application of two different discretization procedures, we have been able to compute the generalization complexity of 20 different benchmark data sets. We have also verified that the relationship, previously obtained only for Boolean input data, between generalization and complexity is preserved, obtaining a clear linear fit (cf. Figure 1). Furthermore, on average, we also found that a better generalization ability can be obtained if a larger number of neurons in the neural architectures is used for more complex functions, even if large deviations can be observed for particular cases. Regarding the two different binarization procedures used, we have found no much difference between their results, even

if we have compared a very simple method (equal width algorithm) to a more complex one using extra information from the data (the CAIM algorithm). For data sets with an unbalanced distribution of output class, some differences in GC values were obtained, implying that in these cases the CAIM algorithm should be the preferred method.

The overall conclusion of the present results should be that in principle the application of the generalization complexity measure to real valued input functions for the problem of finding an adequate size architecture is feasible. We plan to do a more in-depth analysis of the architecture selection method that can be derived from this work, in order to analyze its application in a real practical case.

Acknowledgements

The authors acknowledge support from the Spanish Ministry of Education and Science (MICINN) through grants TIN2008-04985 and from Junta de Andalucía through grants P06-TIC-01615 and P08-TIC-04026 (all including FEDER funds).

References

1. Baum, E.B., Haussler, D.: What Size Net Gives Valid Generalization? Neural Comput. 1(1), 151–160 (1990)
2. Prechelt, L.: A Set of Neural Network Benchmark Problems and Benchmarking Rules. Technical Report 21/94. Fakultat fur Informatik, U. Kalrsruhe, Germany (1994)
3. Barron, A.R.: Approximation and Estimation Bounds for Artificial Neural Networks. Mach. Learn. 14(1), 115–133 (1994)
4. Camargo, L.S., Yoneyama, T.: Specification of Training Sets and the Number of Hidden Neurons for Multilayer Perceptrons. Neural Comput. 13(12), 2673–2680 (2001)
5. Teoh, E., Xiang, C., Chen, K.: Estimating the Number of Hidden Neurons in a Feedforward Network Using the Singular Value Descomposition. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 858–865. Springer, Heidelberg (2006)
6. Mirchandani, G., Cao, W.: On Hidden Nodes for Neural Nets. IEEE Trans. on Circuits and Systems 36(5), 661–664 (1989)
7. Arai, M.: Bounds on the Number of Hidden Units in Binary-Valued Three-Layer Neural Networks. Neural Networks 6(6), 855–860 (1993)
8. Zhang, Z., Ma, X., Yang, Y.: Bounds on the Number of Hidden Neurons in Three-Layer Binary Neural Networks. Neural Networks 16(7), 995–1002 (2003)
9. Yuan, H.C., Xiong, F.L., Huai, X.Y.: A Method for Estimating the Number of Hidden Neurons in Feedforward Neural Networks Based on Information Entropy. In: Computers and Electronics in Agriculture, pp. 57–64 (2003)
10. Liu, Y., Janusz, A., Zhu, Z.: Optimizing the Number of Hidden Neurons in Neural Networks. In: AIAP 2007, Proceedings of the 25th IASTED International Multi-Conference, pp. 121–126 (2007)
11. Haykin, S.: Neural Networks: A Comprehensive Foundation. Macmillan, New York (1994)

12. Gómez, I., Franco, L., Jerez, J.: Neural Network Architecture Selecion. Can Function Complexity Help? Neural Processing Letters 30, 71–87 (2009)
13. Franco, L., Anthony, M.: The Influence of Oppositely Classified Examples on the Generalization Complexity of Boolean Functions. IEEE Transactions on Neural Networks 17, 578–590 (2006)
14. Franco, L.: Generalization Ability of Boolean Functions Implemented in FeedForward Neural Networks. Neurocomputing 70, 351–361 (2006)
15. Franco, L., Cannas, S.A.: Generalization and Selection of Examples in Feedforward Neural Networks. N. Comp. 12(10), 2405–2426 (2000)
16. Franco, L., Cannas, S.A.: Generalization Properties of Modular Networks: Implementing the Parity Function. IEEE Trans. on Neural Networks 12, 1306–1313 (2001)
17. Masters, T.: Practical Neural Network Recipes in C++. Academic Press Professional, Inc., London (1993)
18. Neuralware, Inc.: The Reference Guide (2001)
19. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Francisco (2005)
20. Wong, A.K., Chiu, D.K.: Synthesizing Statistical Knowledge from Incomplete Mixed Mode Data. IEEE Trans. Pattern Analysis and Machine Intelligence 9, 796–805 (1987)
21. Paterson, A., Nibblet, T.B.: ACLS Manual. Int. Teminals Ltd., Edinburgh (1987)
22. Fayyad, U.M., Irani, K.B.: Multi-Interval Discretization of Continuous Valued Attributes for Clasification Learning. In: Proceedings of the 13th International Joint Conference Artificial Intelligence, pp. 1022–1027 (1993)
23. Dougherty, J., Hohavi, R., Sahami, R.: Supervised and Unsupervised Discretization of Continuous Features. In: Proceedings 12th International Conference Machine Learning, pp. 194–202 (1995)
24. Kerber, R.: Chimerge. Discretization of Numeric Attributes. In: Proceedings of Ninth International Conference of Artificial Intelligence, pp. 123–128 (1992)
25. Liu, H., Setiono, R.: Feature Selection Via Discretization. IEEE Knowledge and Data Eng. 9(4), 642–645 (1997)
26. Moller, M.F.: Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. Neural Networks 6(4), 525–533 (1993)
27. Kurgan, L.A., Cios, K.J.: CAIM Discretization Algorithm. IEEE Transactions of Knoweledge and Data Eng. 16(2), 145–153 (2004)
28. Maimon, O., Rokach, L.: Data Mining and Knoweledge Discovery Handbook. Springer, New York (2005)

A New Two-Step Gradient-Based Backpropagation Training Method for Neural Networks

Xuewen Mu[1] and Yaling Zhang[2]

[1] Department of Applied Mathematics, Xidian University, Xi'an, 710071, China
[2] Department of Computer Science, Xi'an Science and Technology University, Xi'an, 710054, China

Abstract. A new two step gradient-based backpropagation training method is proposed for neural networks in this paper. Based on the Barzilai and Borwein steplength update rule and the technique of Resilient Gradient Descent method, we give a new descent direction and steplength update rule. The new two step learning rate improves the speed and the success rate. Experimental results show that the proposed method has considerably improved convergence speed, and for the chosen test problems, outperforms other well-known training methods.

Keywords: Barzilai and Borwein steplength, Resilient Propagation method, Backpropagation training method.

1 Introduction

Artificial neural networks (ANNs) are interdisciplinary information processing techniques rooted in biology, physics, mathematics, and many other fields of science. Primary ANN growth was first witnessed in the early 1940s[1], and widespread research and applications grew in the 1980s. The nonlinearity in engineering, the explosion of data, and the fuzziness of information constitute the basic scientific and technological background for the rapid development of ANN theory and applications in recent years[2, 3].

Among neural networks for static mapping, the error back-propagation neural network is very effective and has widespread application. It consists of a multilayer feed-forward network and is trained by an error back-propagation algorithm. Several adaptive learning algorithms for feedforward neural network have been discovered for solving approximation, pattern recognition, classification and other well known problems[$4-7$]. Many of these algorithms are based on the gradient descent algorithm well known in optimization theory. They usually have a poor convergence rate and depend on parameters which have to be specified by the user, because no theoretical basis for choosing them exists. The values of these parameters are often crucial for the success of algorithm. One of these algorithms is the standard backpropagation(BP)[8]. Although BP is

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 95–101, 2010.

the most common and widely used supervised training algorithm, nevertheless, because of the user depended parameter, it is usually inefficient on large scale problems.

The neural network training can be formulated as a nonlinear unconstrained optimization problem. So the training process can be realized by minimizing the error function E defined by[6, 7]

$$E = \frac{1}{2}\sum_{p=1}^{P}\sum_{j=1}^{N_M}(o_{j,p}^{M} - t_{j,p})^2 = \sum_{p=1}^{P} E_p \tag{1}$$

where$(o_{j,p}^{M} - t_{j,p})^2$ is the squared difference between the actual output value at the $j-$th output layer neuron for pattern p and the target output value. The scalar p is an index over input-output pairs. The general purpose of the training is to search an optimal set of connection weights in the manner that the errors of the network output can be minimized.

The BP algorithm uses the steepest descent search direction with a fixed step size α in order to perform the minimization of the error function. The iterative form of this algorithm is

$$w^{k+1} = w^k - \eta_k g^k \tag{2}$$

where w denotes a column weight vector with components $w_1, w_2, \cdots, w_n$ which is defined in the $n-$ dimensional real space R^n, and g the gradient vector of the error function E at w, that is $g = \nabla E(w)$. E represents the batch error measure defined as the sum of squared differences error function over the entire training set.

The inefficiency of steepest descent is due to the fact that the minimization directions and step sizes are chosen poorly; if the first step size does not lead directly to the minimum, steepest descent will zig-zag with many smal steps[6, 7].

There are a number of modifications and extensions of the standard algorithm to the point of fundamental changes of the gradient descent by for example conjugate gradient and Newton methods. Beyond this there are also a number of alternative training methods based on general optimization techniques, such as stochastic methods[9] or Genetic Algorithms (GA) [10].

In 1988 Barzilai and Borwein [11] proposed a gradient descent method (BB method) that uses a different strategy for choosing the step length. This is based on an interpretation of the quasi-Newton methods in a very simple manner. The steplength along the negative gradient direction is computed from a two-point approximation to the secant equation from quasi-Newton methods. In [11] Barzilai and Borwein proved that for the two-dimensional quadratic case the BB method is R-superlinear convergent.

A new two step gradient-based backpropagation training method is proposed for neural networks in this paper. Based on the Barzilai and Borwein steplength update rule and the technique of Resilient Gradient Descent method, we give a new descent direction and steplength update rule. The new two step learning rate improves the speed and the success rate. Experimental results show that

the proposed method has considerably improved convergence speed, and for the chosen test problems, outperforms other well-known training methods.

The plan of this paper is as follows. In section 2 we present some classic training algorithms. In section 3 we present the proposed training algorithm. Section 4 contains our numerical examples and results. The final section contains a discussion of our results, and some concluding remarks.

2 The Barzilai and Borwein Method

The main idea of Barzilai and Borweins approach[11] is to use the information in the previous interation to decide the step-size in the current iteration. The iteration (2) is viewed as

$$w^{k+1} = w^k + d^k \tag{3}$$

where $d^k = -H_k g^k = -H_k \nabla E(w^k)$, $H_k = \eta_k I$. In order to force the matrix H_k to have certain quasi-Newton property, it is reasonable to require either

$$\min ||s^{k-1} - H_k y^{k-1}||^2 \tag{4}$$

or

$$\min ||H_k^{-1} s^{k-1} - y^{k-1}||^2 \tag{4'}$$

where $s^{k-1} = w^k - w^{k-1}$ and $y^{k-1} = g^k - g^{k-1}$, because the quasi-Newton matrix H_k satisfies the condition

$$s_{k-1} = H_k y_{k-1} \tag{5}$$

Now, from $H_k = \eta_k I$ and relations (4)-(4') we can obtain two step-sizes:

$$\eta_k = \frac{< s^{k-1}, y^{k-1} >}{||y^{k-1}||_2^2} \tag{6}$$

and

$$\eta_k = \frac{||s^{k-1}||_2^2}{< s^{k-1}, y^{k-1} >} \tag{6'}$$

respectively. For convex quadratic functions in two variables, Barzilai and Borwein [11] shows that the gradient method (2) with η_k given by (6) converges R-superlinearly and R-order is $\sqrt{2}$.

The step size contains second order information without estimating the Hessian matrix. The results show that this choice of the step size is very efficient[11].

3 The Two Step Gradient-Based Backpropagation Training Method

Based on the Barzilai and Borwein steplength update rules and the technique of Resilient Gradient Descent method, we give a new descent direction and steplength update rule to improve the speed and the success rate.

3.1 The Basic Idea of the Proposed Method

The classic iteration form of unconstrained optimization methods is viewed as

$$w^{k+1} = w^k + \beta_k d^k \tag{7}$$

In fact, Barzilai and Borwein steplength update also can be see a descent direction with a fixed stepsize which, it is that $d^k = -\eta_k g^k, \beta_k = 1$.

Barzilai and Borwein steplength contains second order information without estimating the Hessian matrix. Based on the relation (5), let quasi-Newton matrix

$$H_k = Diag(\alpha^k) \tag{8}$$

where $Diag(\alpha^k)$ denote a diagonal matrix, whose diagonal elements are the elements of row vector α^k. Based on relation (6), we obtain a new stepsize update rule

$$\alpha_i^k = \frac{s_i^{(k-1)}}{y_i^{(k-1)}}$$

Because the direction d^k is not a descent direction if the parameter $\alpha_i^k < 0$, so we modify the parameter as follows

$$\alpha_i^k = max\{\theta_0, min\{\theta_1, \alpha_i^k\}\} \tag{9}$$

where the constant θ_0 and θ_1 are the lower limit and the upper limit for η_k. we set their values $\theta_0 = 0.001, \theta_1 = 5$.

Due to the commonly used semi-linear transfer functions (sigmoid, hyperbolic tangent) and their actually desired property to compress an infinite input range into a finite output range by the fact that their slope approaches zero as the input moves towards 1, the gradient can have a very small magnitude, even though the weights are far away from their optimal values. Therefore, Resilient Gradient Descent(RProp) [12] uses only the sign (plus an externally defined value) instead of the signed magnitude of the current slope to update the weights. This way the algorithm requires relatively few resources and is rather fast to obtain any solution. If very good quality (low network error) is desired, its performance usually deteriorates.

Based on the update technique of the Resilient Gradient Descent method[12], we give a stepsize update method according to the following learning-rule:

$$\beta_k = \begin{cases} \beta^+, & if \frac{\partial E_{k-1}}{\partial E_{ij}} * \frac{\partial E_k}{\partial E_{ij}} > 0 \\ \beta^-, & if \frac{\partial E_{k-1}}{\partial E_{ij}} * \frac{\partial E_k}{\partial E_{ij}} < 0 \\ 1, & else \end{cases} \tag{10}$$

where β^+, β^- are increasing and decreasing factors, which satisfy $0 < \beta^- < 1 < \beta^+$.

Relation (8) contains more second order information than $H_k = \eta_k I$. It is very important to choose a good stepsize. Experimental results show that the proposed method has considerably improved convergence speed for the chosen test problems.

3.2 The New Two Step Gradient-Based Backpropagation Method

In this section, we give a new two step gradient-based backpropagation training method for neural network.

The New Two Step Gradient-based Backpropagation Method (NGBM)

Step 1: Initialize by setting the number of epochs $k = 0$, the weights w^0, the learning rate to an arbitrary value $\eta^0 = 0.1$, and the values of upper and lower limits $\theta_0 = 0.001, \theta_1 = 5$, the error goal ϵ.

Step 2: Compute the weight vector w^1 according to relation (2) and set $k = k + 1$.

Step 3: Compute the new parameter α_i^k using the relation (9).

Step 4: Update the weight vector w_{k+1} according to the relation (10).

Step 5: Check if $E(w_{k+1}) > \epsilon$, set k = k + 1 and go to Step 3; otherwise get the final weight vector w^*, and the corresponding value of E.

4 Numerical Results

The numerical simulation has been developed to study the performance of the new two step gradient-based backpropagation training method. The simulations have been carried out on a Pentium 1.0GHz PC Dell computer using Visual C++.NET 2003. The performance of the $NGBM$ algorithm has been evaluated and compared with the batch versions of BP, Barzilai and Borwein method (BB), and scaled conjugate gradient method (SCG) [13].

Three problems have been tested, which include the eXclusive OR Problem, the 3-bit Parity Problem, and the Continuous Function Approximation Problem. We test the run time of each algorithm for the three problems.

4.1 The Exclusive-OR Problem

The first problem we have test is the eXclusive - OR (XOR) Boolean function problem[7], which is considered as a classical problem for the FNN training. The XOR function maps two binary inputs to a single binary output. As it is well known this function is not linearly separable. The selected architecture of the FNN is 2-2-1 (six weights and three biases) with logistic neurons with biases in the layers . The error goal has been set to 0.01 and the maximum epochs to 10000. For the BP algorithms the learning rate is chosen to be 0.1. The results of the simulations are presented in Table 1.

Table 1. The XOR Problem

algorithms	BP	BB	SCG	NGBM
times(ms)	2906	2890	2010	859

4.2 The 3-Bit Parity Problem

In this simulation we test the 3-bit parity problem[7], which can be considered as the 3-bit version of the XOR problem. This problem maps three binary inputs to a single binary output. The target of the output is 1, if the number of 1 bits in the input is odd, and 0 otherwise. The selected architecture of the FNN is 3-4-1 (sixteen weights and three biases) with logistic neurons with biases in the layers. The error goal has been set to 0.01 and the maximum epochs to 10000. For the BP the learning rate is chosen to be 0.1 instead of the default value 0.01 to accelerate their convergence. The results of the simulations are presented in Table 2.

Table 2. The 3-bit Parity Problem

algorithms	BP	BB	SCG	NGBM
times(ms)	27093	14953	8593	6640

4.3 The Continuous Function Problem

The last problem is the approximation of the continuous function $F(x) = \frac{1}{2} sin(\frac{1}{2}\pi x) + \frac{1}{2}$. This problem maps one real input to a single real output. The input values are 20 equally spaced points $x_i \in [0, 19/20]$ and the target values are the mapping of these points from function $F(x)$. As it is cleared, we have 20 patterns and each pattern is consisted of one input $x_i \in [0, 19/20]$ and one target value $F(x)$. The selected architecture of the FNN is 1-10-1 (twenty weights and eleven biases) with logistic neurons with biases in the layers. The error goal has been set to 0.1 and the maximum epochs to 10000. The results of the simulations are presented in Table 3.

Experimental results show that the proposed method needs less time than the other three method. So the method is fast, simple, and efficient.

Table 3. The Continuous Function Problem

algorithms	BP	BB	SCG	NGBM
times(ms)	2078	1031	296	289

5 Conclusions

A new two step gradient-based backpropagation training method is proposed. It is shown that this method improves the convergence speed in several classical test problems. Our experimental results clearly show that the proposed method outperforms the classical training algorithms (BP, BB and SCG). It runs much faster, has improved average number of epochs, and better convergence rates.

Acknowledgments. The author would like to thank the reviewer for his advices. The present work was supported by National Science Basic Research Plan in ShaanXi Province of China(Program No. SJ08A10).

References

1. McCulloch, W.S., Pitts, W.: A Logical Calculus of the Ideas Immanent in Nervous Activity. Bull. of Math. Bio. 1(5), 115–133 (1943)
2. Liu, Z., Liu, J.: Seismic-controlled Nonlinear Extrapolation of Well Parameters Using Neural Networks. Geophysics 63(6), 2035–2041 (1998)
3. Seiffert, U.: Training of large-Scale feed-forward neural networks. In: 2006 International Joint Conference on Neural Networks, pp. 10780–10785 (2006)
4. Hammer, B., Strickert, M., Villmann, T.: Supervised neural gas with general similarity measure. Neural Processing Letters 21, 21–44 (2005)
5. Bishop, C.M.: Neural Networks for Pattern Recognization. Clarendon Press, Oxford (1995)
6. Plagianakos, V.P., Sotiropoulos, D.G., Vrahatis, M.N.: A Nonmonotone Backpropagation Training Method for Neural Networks. Dept. of Mathematics, Univ. of Patras, Technical Report No. 98-04 (1998)
7. Sotiropoulos, D.G., Kostopoulos, A.E., Grapsa, T.N.: A spectral version of perry's conjugate gradient method for neural network training. In: Proceedings of 4th GRACM Congress on Computational Mechanics, pp. 291–298 (2002)
8. Lippmann, R.P.: An introduction to Computing With Neural Nets. IEEE ASSP Magazine 4(87), 4–23 (1987)
9. Seiffert, U., Michaelis, B.: Directed random search for multiple layer perceptron training. In: Neural Networks for Signal Processing IEEE, pp. 193–202 (2001)
10. Seiffert, U.: Multiple layer perceptron training using Genetic Algorithms. In: Proceedings of the 9. European Symposium on Artificial Neural Networks, pp. 159–164 (2001)
11. Barzilai, J., Borwein, J.M.: Two point step size gradient methods. IMA J. Numer. Anal. 8, 141–148 (1988)
12. Riedmiller, M., Braun, H.: A direct adaptive method for faster Backpropagation learning: The RPROP algorithm. In: Proceedings of the IEEE International Conference on Neural Networks, pp. 586–591. IEEE, New Jersey (1993)
13. Moeler, M.F.: A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks 6, 525–533 (1993)

A Large-Update Primal-Dual Interior-Point Method for Second-Order Cone Programming

Liang Fang[1], Guoping He[2], Zengzhe Feng[3], and Yongli Wang[2]

[1] College of Mathematics and System Science, Taishan University, 271021, Tai'an, P.R. China
fangliang3@163.com
[2] College of Information Science and Engineering, Shandong University of Science and Technology, 266510, Qingdao, P.R. China
[3] College of Information Engineering, Taishan Medical University, 271000, Tai'an, P.R. China

Abstract. A large-update primal-dual interior-point algorithm is presented for solving second order cone programming. At each iteration, the iterate is always following the usual wide neighborhood $\mathcal{N}_\infty^-(\tau)$, but not necessary staying within it. However, it must stay within a wider neighborhood $\mathcal{N}(\tau,\beta)$. We show that the method has $O(\sqrt{r}L)$ iteration complexity bound which is the best bound of wide neighborhood algorithm for second-order cone programming.

Keywords: Second-order cone programming; Smoothing method; Interior-point method; Q-quadratic convergence; Central path.

1 Introduction

Second-order cone programming (SOCP) is to minimize or maximize a linear function over the intersection of an affine space with the Cartesian product of a finite number of second-order cones (SOCs). It has various applications in many fields, such as engineering, support vector machine, combinatorial optimization and so on [1,2,6,8]. Moreover, it turns out that interior point methods (IPMs) can solve SOCP efficiently [4,5]. Many IPMs for SOCP can be found e.g., in [1,9]. Most of them are concentrated on primal-dual IPMs.

The IPMs choose a target on the central path and apply the Newton method to move closer to the target at each iteration, while confining the iterate to stay within a certain neighborhood of the analytic central path. In the first case the method is called a small-update method. It uses full Newton steps and the iterates stay in a small neighborhood of the central path, known as the $\mathcal{N}_2$-neighborhood. The worst case iteration bound was proved to be $O(\sqrt{r}L)$. The second method is that the iterates are allowed to move in a wide neighborhood of the central path, known as $\mathcal{N}_\infty^-$. The worst case iteration bound was proved to be $O(rL)$. However, in practice, large-update IPMs are much more efficient than small-update IPMs [3,7]. To the author's knowledge, in the context of path-following approach, none had succeeded in retaining the $O(\sqrt{r}L)$ complexity

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 102–109, 2010.

while allowing a large update of the target along the central path at all iterates. Thus, there is a gap between the practical behavior of IPMs and their theoretical performance. The main motivation of the paper is to narrow this gap between theory and practice for large-update methods.

The paper is organized as follows. In the next section, we introduce the SOCP problem and the new wide neighborhood. In section 3, we describe the algorithm based on the new wide neighborhood and the key result that presents the low computational complexity bound. Some technical results and iteration complexity bound are given in section 4.

The following notations are used throughout this paper. The superscript ^{T} denotes transpose. $\mathcal{R}^m$ denotes the m-dimensional Euclidean space. The set $\mathcal{Q}_{n_i}$ is the SOC with dimension $n_i \geq 1$, $i = 1, \ldots, r$, which is defined as

$$\mathcal{Q}_{n_i} = \{(x_0^i; \bar{x}^i) \in \mathcal{R} \times \mathcal{R}^{n_i-1} : {x_0}^i \geq \|\bar{x}^i\|\},$$

where $\|\cdot\|$ represents the general Euclidean norm, and $\bar{x}^i = (x_1^i, \cdots, x_{n_{i-1}}^i)^T \in \mathcal{R}^{n_i-1}$. Let $x = (x^1; \cdots; x^r)$ where $x^i = (x_0^i; \bar{x}^i) \in \mathcal{R} \times \mathcal{R}^{n_i-1}, n_1 + \cdots + n_r = n$. For $x \in \mathcal{Q}$, we mean $x^i \in \mathcal{Q}_{n_i} \subseteq \mathcal{R}^{n_i}, i = 1, \cdots, r$. For simplicity, we use "," for adjoining vectors or matrices in a row and ";" for adjoining them in a column. The set of all $m \times n$ matrices with real entries is denoted by $\mathcal{R}^{m\times n}$. For symmetric matrix $A \in \mathcal{R}^{n\times n}, A \succeq 0$ means A is positive semidefinite and $A \succ 0$ means A is positive definite. Similarly, $x \succeq 0$ means $x \in \mathcal{Q}$ and $x \succ 0$ means $x \in \text{int}\mathcal{Q}$. The trace of a vector $x^i = ({x_0}^i; \bar{x}^i) \in \mathcal{R}^{n_i}$ is denoted by $\text{tr}x^i = \lambda_1^i + \lambda_2^i = 2x_0^i$, where $\lambda_1^i = x_0^i - \|\bar{x}^i\|$ and $\lambda_2^i = x_0^i + \|\bar{x}^i\|$ are the eigenvalues of x^i. We also define determinant of x^i by $\det(x^i) = \lambda_1^i\lambda_2^i = (x_0^i)^2 - \|\bar{x}^i\|^2$. Given $x = (x_0; \bar{x}), s = (s_0; \bar{s}) \in \mathcal{R}\times\mathcal{R}^{n-1}$, we define $\langle x, s\rangle = \text{tr}(x \circ s)$, where "$\circ$"is a bilinear map defined by $x \circ s = (x^1 \circ s^1; \cdots; x^r \circ s^r)$, where $x^i \circ s^i = ((x^i)^T s^i; x_0^i\bar{s}^i + s_0^i\bar{x}^i)$. From the definition, we can easily see that $\langle x^i, s^i\rangle = 2(x^i)^T s^i$. The Frobenius norm of $x^i \in \mathcal{Q}_{n_i}$ is defined by $\|x^i\|_F = \sqrt{(\lambda_1^i)^2 + (\lambda_2^i)^2} = \sqrt{\text{tr}((x^i)^2)} = \sqrt{\langle x^i, x^i\rangle} = \sqrt{2}\|x^i\|$. For any $a \in \mathcal{R}$, a^+ denotes its nonnegative part, i.e., $a^+ := \max\{a, 0\}$, and a^- denotes its nonpositive part, i.e., $a^- := \min\{a, 0\}$. It is evident that $a^+ \geq 0, a^- \leq 0$, and $a = a^+ + a^-$. For a vector $x^i \in \mathcal{R}^{n_i}$, its eigenvalue factorization is $x^i := \lambda_1^i u_1^i + \lambda_2^i u_2^i$, where $\lambda_j^i = x_0^i + (-1)^j\|\bar{x}^i\|$,

$$u_j^i = \begin{cases} \frac{1}{2}(1; (-1)^j \frac{\bar{x}}{\|\bar{x}^i\|}), & x^i \neq 0; \\ \frac{1}{2}(1; (-1)^j\omega), & x^i = 0, \end{cases} \tag{1}$$

for $j = 1, 2$, with any $\omega \in \mathcal{R}^{n_i-1}$ such that $\|\omega\| = 1$. If $\bar{x}^i \neq 0$, then above decomposition is unique. We use $(x^i)^+$ and $(x^i)^-$ to denote the following vectors

$$(x^i)^+ := (\lambda_1^i)^+ u_1^i + (\lambda_2^i)^+ u_2^i, \; (x^i)^- := (\lambda_1^i)^- u_1^i + (\lambda_2^i)^- u_2^i,$$

respectively. It is evident that $(x^i)^+ \succeq 0, (x^i)^- \preceq 0$ and $x^i = (x^i)^+ + (x^i)^-$.

2 SOCP Problem and the New Wide Neighborhood

We consider the SOCP

$$\min\{c^T x : Ax = b, x \in \mathcal{Q}\} \tag{2}$$

and its dual problem

$$\max\{b^T y : A^T y + s = c, s \in \mathcal{Q}, y \in \mathcal{R}^m\}, \tag{3}$$

where $A = (A_1, \ldots, A_r) \in \mathcal{R}^{m\times n}, c = (c^1; \ldots; c^r) \in \mathcal{R}^n, b \in \mathcal{R}^m$ are given data, $\mathcal{Q} = \mathcal{Q}_{n_1} \times \cdots \times \mathcal{Q}_{n_r}$, $x = (x^1; \ldots; x^r)$ and $s = (s^1; \ldots; s^r) \in \mathcal{Q}$ are the primal and dual variables respectively. The vector $e = (e^1; \cdots; e^r)$ is the identity element of $\mathcal{Q}$, where $e^i = (1; 0) \in \mathcal{R} \times \mathcal{R}^{n_i-1}$.

For any $x \in \mathcal{R}^n$, we have the conclusion: $x \succeq 0 \Leftrightarrow x^T s \geq 0$, for all $s \in \mathcal{Q}$.

Throughout the paper, we make the following assumptions.

Assumption 1. Both (2) and (3) are strictly feasible.

Assumption 2. The row vectors of matrix A are linearly independent.

It is well known that under Assumption 1 and 2, the SOCP is equivalent to its optimality conditions (see [2])

$$Ax = b, \quad A^T y + s = c, \quad x \circ s = 0, \quad x, s \in \mathcal{Q}, y \in \mathcal{R}^m. \tag{4}$$

The central path consists of points $(x(\mu), y(\mu), s(\mu))$ satisfying the perturbed optimality conditions (PC_μ)

$$Ax = b, \quad A^T y + s = c, \quad x \circ s = \mu e, \; x, s \in \mathcal{Q}, \; y \in \mathcal{R}^m. \tag{5}$$

It is also well known that under Assumptions 1 and 2 the solution $(x(\mu), y(\mu), s(\mu))$ of system (5) exists and it is unique for each $\mu > 0$. Moreover, the limit $(x^*, y^*, s^*) = \lim_{\mu\downarrow 0}(x(\mu), y(\mu), s(\mu))$ exists and it is a primal-dual optimal solution, i.e., x^* and (y^*, s^*) are solutions of (2) and (3), respectively.

IPMs follow the solutions to (PC_μ) as μ goes to zero. Note that the duality gap of the solutions is proportional to μ, since $\langle x, s\rangle = \mathrm{tr}(x \circ s) = \mu \mathrm{tr}(e) = 2r\mu$. IPMs employ Newton's method to target the solution of (7), where $\sigma \in (0, 1)$, (x, y, s) is the current iterate, and $\mu = x^T s/r$.

Suppose that the point (x, y, s) is strictly feasible, i.e., $x \succ 0$ and $s \succ 0$. Newton's method amounts to linearizing the system (5), thus yielding the system

$$A\Delta x = 0, \quad A^T \Delta y + \Delta s = 0, \quad s \circ \Delta x + x \circ \Delta s = \mu e - x \circ s, \tag{6}$$

which gives the usual search directions of feasible primal-dual IPMs for SOCP.

Note that above system (6) might not be well defined if its Jacobian matrix is singular. To obtain a Newton-type system that has a unique solution, people usually use some scaling schemes.

For any $x = (x_0; \bar{x}) \in \mathcal{R}^n$, we define the arrow-shaped matrix as follows

$$L(x) = \begin{pmatrix} x_0 & \bar{x}^T \\ \bar{x} & x_0 I \end{pmatrix},$$

where I represents the $(n-1) \times (n-1)$ identity matrix.

It is easy to verify that $x \circ s = L(x)s = L(s)x = L(x)L(s)e$ for any $x, s \in \mathcal{R}^n$. Moreover, $L(x) \succ 0$ if and only if $x \succ 0$.

We say two elements x and y of a Jordan algebra $\mathcal{J}$ operator commute if $L(x)L(y) = L(y)L(x)$. Moreover, for any $x = (x_0; \bar{x}) \in \mathcal{R}^n$, the quadratic representation associated with x is defined as

$$Q_x = 2L(x)^2 - L(x^2) = \begin{pmatrix} \|x\|^2 & 2x_0\bar{x}^T \\ 2x_0\bar{x} & \det(x)I + 2\bar{x}\bar{x}^T \end{pmatrix} = 2xx^T - \det(x)F,$$

where $F = \mathrm{diag}(1, -1, \cdots, -1)$ is reflection matrix.

$L(x)$ and Q_x have the following properties (see [1] and [2]).

Property 1. Let x has the spectral decomposition $x = \lambda_1 u_1 + \lambda_2 u_2$.
(1) Q_x and $L(x)$ commute and thus share a system of eigenvectors.
(2) Q_x is nonsingular if and only if x is nonsingular.
(3) If $x \succeq 0$, then x is nonsingular if and only if $L(x)$ is nonsingular.

Property 2. For each $x, y \in \mathcal{R}^n$, x nonsingular, $\alpha \in \mathcal{R}$, and integer t,
(1) $Q_x x^{-1} = x$ and thus $Q_x^{-1}x = x^{-1}$; (2) $Q_{x^{-1}} = Q_x^{-1}$;
(3) if y is also nonsingular, then $(Q_x y)^{-1} = Q_{x^{-1}} y^{-1}$;
(4) $\det(Q_x y) = \det^2(x)\det(y)$; (5) $\langle x, s\rangle = \langle Q_p x, Q_{p^{-1}} s\rangle$;
(6) $\langle x \circ y, z\rangle = \langle x, y \circ z\rangle$; (7) if $p \succ 0$ and $x \succ 0$, then $Q_p x \succ 0$.

Lemma 1. *If $\mathcal{J}$ is a simple Jordan algebra, $x, s \in \mathcal{J}$ and $x, s \succ 0$, p is a scaling, $\mu \neq 0$, then $x \circ s = \mu e \Longleftrightarrow Q_p x \circ Q_{p^{-1}} s = \mu e \Longleftrightarrow Q_{p^{-1}} x^{-1} \circ Q_p s^{-1} = \mu^{-1} e$.*

For a scaling p, (PC_μ) can be equivalently written as

$$\tilde{A}\tilde{x} = b, \quad \tilde{A}^T y + \tilde{s} = \tilde{c}, \quad \tilde{x} \circ \tilde{s} = \mu e, \ \tilde{x}, \tilde{s} \in \mathcal{Q}, y \in \mathcal{R}^m, \tag{7}$$

where $\tilde{A} = AQ_{p^{-1}}, \tilde{x} = Q_p x, \tilde{s} = Q_{p^{-1}} s$ and $\tilde{c} = Q_{p^{-1}} c$.

Using the linear transformation Q_p, we scale x and s by

$$\tilde{x} = Q_p x, \tilde{s} = Q_{p^{-1}} s = Q_p^{-1} s. \tag{8}$$

For $p = x^{-1/2}$ we get the xs-method, for $p = s^{1/2}$ we get the sx-method, and for the choice of $p = [Q_{x^{1/2}}(Q_{x^{1/2}} s)^{-1/2}]^{-1/2} = [Q_{s^{-1/2}}(Q_{s^{1/2}} x)^{1/2}]^{-1/2}$ we get the Nesterov-Todd (NT) method. The scaled Newton equations are stated by

$$\tilde{A}\Delta\tilde{x} = 0, \quad \tilde{A}^T \Delta y + \Delta\tilde{s} = 0, \quad \tilde{s} \circ \Delta\tilde{x} + \tilde{x} \circ \Delta\tilde{s} = \sigma\mu e - \tilde{x} \circ \tilde{s}, \tag{9}$$

where $\sigma \in [0, 1]$ is a central parameter.

Lemma 2. *Let x, s, p be all positive definite, $\tilde{x}$ and $\tilde{s}$ be defined by (8), then*
(1) the vectors $Q_{x^{1/2}} s$ and $Q_{s^{1/2}} x$ have the same spectrum;
(2) the vectors $Q_{x^{1/2}} s$ and $Q_{\tilde{x}^{1/2}} \tilde{s}$ have the same spectrum.

Lemma 3. *$L(x)$ is a self-adjoint operator, i.e., for any $x, y, z \in \mathcal{R}^n$, we have $\langle L(x)y, z\rangle = \langle y, L(x)z\rangle$. Furthermore, $Q(x)$ is also a self-adjoint operator.*

For the purpose of scaling, we define the transformations $Q_p = Q_{p^1} \oplus \cdots \oplus Q_{p^r} = \mathrm{diag}(Q_{p^1}, \cdots, Q_{p^r})$ with respect to $p^i \succ 0, i = 1, \cdots, r$.

The normal so-called wide neighborhood is defined as $\mathcal{N}_\infty^-(\tau) = \{(x, y, s) \in \mathcal{F}^0(P) \times \mathcal{F}^0(D) | \lambda_{\min}(w) \geq \tau\mu\}$, while our algorithm will restrict the iterates to the new wide neighborhood

$$\mathcal{N}(\tau, \beta) = \{(x, y, s) \in \mathcal{F}^0(P) \times \mathcal{F}^0(D) \mid \|(\tau\mu e - w)^+\|_F \leq \beta\tau\mu\}, \tag{10}$$

where $\mu = x^T s/r$, $w = Q_p s$, and we choose $p = x^{1/2}$ throughout the paper.

Note that, if $(x, y, s) \in \mathcal{N}_\infty^-(\tau)$, then $\tau\mu e - w \preceq 0$, namely $\|(\tau\mu e - w)^+\|_F = 0$. Therefore, $(x, y, s) \in \mathcal{N}(\tau, \beta)$. Thus, $\mathcal{N}(\tau, \beta) \supseteq \mathcal{N}_\infty^-(\tau)$. In other words, the neighborhood $\mathcal{N}(\tau, \beta)$ is wider than $\mathcal{N}_\infty^-(\tau)$, where $\beta \in (0, 1/2]$, $\tau \in (0, 1]$.

The following lemmas give the symmetry and scale-invariance of $\mathcal{N}(\tau, \beta)$.

Lemma 4. *The neighborhood $\mathcal{N}(\tau, \beta)$ is symmetric, i.e., for any $(x, s) \in \mathcal{R}^n \times \mathcal{R}^n$, we have $\|(\tau\mu e - Q_{x^{1/2}} s)^+\| = \|(\tau\mu e - Q_{s^{1/2}} x)^+\|$.*

Proof. The result can be easily obtained by the first statement of Lemma 2.

Lemma 5. *The neighborhood $\mathcal{N}(\tau, \beta)$ is scaling invariant, that is,*

$$(x, y, s) \in \mathcal{N}(\tau, \beta) \Longleftrightarrow (\tilde{x}, y, \tilde{s}) \in \mathcal{N}(\tau, \beta).$$

Proof. By the second statement of Lemma 2, $Q_{x^{1/2}} s$ has the same spectrum as $Q_{s^{1/2}} x$. Therefore, $\tau\mu e - Q_{x^{1/2}} s$ and $\tau\mu e - Q_{s^{1/2}} x$ have the same spectrum.

3 Description of the Algorithm

Suppose that the current iterate is $z := (x, y, s) \in \mathcal{N}(\tau, \beta)$, define the vector

$$v := [(\tau\mu e - w)^+]^2 - \delta^2[(\tau\mu e - w)^-]^2 + w, \tag{11}$$

where

$$\delta = -\frac{\|(\tau\mu e - w)^+\|_F}{\|(\tau\mu e - w)^-\|_F}, \tag{12}$$

such that $\langle v, e\rangle = 2r\mu$, $w = Q_p s, p = x^{1/2}$, and $\mathcal{N}(\tau, \beta)$ is defined by (10).

Proposition 1. *If $z := (x, y, s) \in \mathcal{N}(\tau, \beta)$, and v is defined by (11),then*

$$(i)\ v \succeq \tau\mu e; \quad (ii)\ \|\delta(\tau\mu e - w)^-\|_F^2 = \|(\tau\mu e - w)^+\|_F^2;$$
$$(iii)\ \|(w)^{-1/2} \circ (v - w)\|_F^2 \leq (\beta\tau\mu)^3 \ \textit{when}\ 0 < \beta \leq \tfrac{1}{1+4r}, r \geq 1.$$

Proof. (i)and (ii) can be easily obtained from the definition of v, δ and $\mathcal{N}(\tau, \beta)$.

(iii) From $z \in \mathcal{N}(\tau,\beta)$, we have $w \succeq (1-\beta)\tau\mu e$. Therefore, by (12) we have

$$\begin{aligned}
\|(w)^{-1/2} \circ (v-w)\|_F^2 &= \|(w)^{-1/2} \circ \left\{[(\tau\mu e - w)^+]^2 - \delta^2[(\tau\mu e - x\circ s)^-]^2\right\}\|_F^2 \\
&\le \|(w)^{-1/2}\|_F^2 \|[(\tau\mu e - w)^+]^2 - \delta^2[(\tau\mu e - x\circ s)^-]^2\|_F^2 \\
&\le 2r/[(1-\beta)\tau\mu]\left\{[\|(\tau\mu e - w_{xs})^+]^2\|_F^2 + [\|\delta(\tau\mu e - w)^-]^2\|_F^2]\right\} \\
&\le 2r/[(1-\beta)\tau\mu]\left\{[\|(\tau\mu e - w)^+]\|_F^4 + [\|\delta(\tau\mu e - w)^-]\|_F^4]\right\} \\
&= \frac{4r}{(1-\beta)\tau\mu}\|(\tau\mu e - w)^+\|_F^4 \le \frac{4r}{(1-\beta)\tau\mu}\beta^4\tau^4\mu^4 \\
&= \frac{4r}{(1-\beta)}\beta^4\tau^3\mu^3 \le (\beta\tau\mu)^3,
\end{aligned}$$

where the first inequality is obtained by triangle inequality, the fact

$$\|(w)^{-1/2}\|_F^2 = tr[(w)^{-1}] = \sum_{i=1}^{2r}\frac{1}{\lambda_i(w_{xs})} \le \frac{2r}{(1-\beta)\tau\mu}$$

and $(\tau\mu e - w)^+ \circ (\tau\mu e - w)^- = 0$. The proof is completed.

In order to derive the Newton direction and state our primal-dual wide neighborhood IPM, we consider the scaled Newton equations:

$$\tilde{A}\Delta\tilde{x} = 0, \quad \tilde{A}^T\Delta y + \Delta\tilde{s} = 0, \quad \Delta\tilde{x} + Q_{x^{1/2}}\Delta\tilde{s} = t\tilde{v} - \tilde{x}, \tag{13}$$

where $\Delta z := (\Delta x, \Delta y, \Delta s) \in \mathcal{R}^n \times \mathcal{R}^m \times \mathcal{R}^n$ is the search direction, and $t \in [0,1]$ is the unknown parameter. The new iterate point can be written as $\bar{x} = x + \Delta x, \bar{y} = y + \Delta y, \bar{s} = s + \Delta s$, or $\bar{z} = z + \Delta z$.

We are now in the position to describe our new algorithm.

Algorithm 3.1 (A wide neighborhood IPM for SOCP.)

Step 0. Input $\varepsilon > 0$, $0 < \tau \le 1$, $t_0 \in [0,1]$, $0 < \beta \le \frac{1}{1+4r}$, $z^0 := (x^0, y^0, s^0) \in \mathcal{N}(\tau,\beta)$. Compute $\mu_0 = \langle x^0, s^0\rangle$, and set $k := 0$.

Step 1. If $\langle x^k, s^k\rangle \le \varepsilon$, then stop.

Step 2. Compute $v = v^k$ by (11), and solve $\Delta\tilde{z}(t_k) = (\Delta\tilde{x}(t_k), \Delta y(t_k), \Delta\tilde{s}(t_k))$ from (13). Let $\Delta z(t_k) = (Q_{x^{-1/2}}\Delta\tilde{x}(t_k), \Delta y(t_k), Q_{x^{1/2}}\Delta\tilde{s}(t_k))$.

Step 3. Let $\bar{t_k}$ be the smallest t such that $z(t_k) \in \mathcal{N}(\tau,\beta)$, for any $t \in [\bar{t}, 1]$.

Step 4. Choose $t_k = \bar{t}$. Compute $z^{k+1} = z^k + \Delta z(t_k)$ and let $\mu_{k+1} = t_k\mu_k$. Set $k := k+1$, and go to step 1.

4 Technical Results and Iteration Complexity Bound

In order to obtain desired conclusions, we need the following important results:

Lemma 6. *Suppose that the $v, z, \tilde{z}$ is defined as above, then it holds that*

$$(i)\bar{x}\circ\bar{s} = tv + \Delta x\circ\Delta s; (ii)\langle\Delta x, \Delta s\rangle = 0; (iii)\bar{\mu} = \frac{\langle\bar{x},\bar{s}\rangle}{2r} = t\mu \tag{14}$$

Proof. (i) By (13), a direct calculation yields

$$\begin{aligned}\bar{x} \circ \bar{s} &= (x + \Delta x) \circ (s + \Delta s) = x \circ s + x \circ \Delta s + \Delta x \circ s + \Delta x \circ \Delta s \\ &= x \circ s + tv - x \circ s + \Delta x \circ \Delta s = tv + \Delta x \circ \Delta s.\end{aligned}$$

(ii) From Assumption 1 and the fact that (x, y, s) is a primal-dual feasible solution, we can easily conclude that $\Delta x^T \Delta s = 0$. Hence, $\langle \Delta x, \Delta s \rangle = 2\Delta x^T \Delta s = 0$.
(iii) Using (i), (ii), we obtain

$$\bar{\mu} = \frac{\langle \bar{x}, \bar{s} \rangle}{2r} = \frac{\mathrm{tr}(tv)}{2r} + \frac{\mathrm{tr}(\Delta x \circ \Delta s)}{2r} = \frac{t\mathrm{tr}(v)}{2r} = \frac{t\langle v, e \rangle}{2r} = \frac{2rt\mu}{2r} = t\mu.$$

Lemma 7. *Suppose that $h, q \in \mathcal{R}^n$, $\langle h, q \rangle \geq 0, h + q = d$, then $\|(h \circ q)^+\|_F \leq \frac{1}{4}\|d\|_F^2$.*

Proof. It follows from $\gamma^2 = (h + q)^2 = h^2 + q^2 + 2(h \circ q) \geq 4(h \circ q)$ that $\|(h \circ q)^+\|_F \leq \frac{1}{4}\|d\|_F^2$.

Lemma 8. *For any $u, v \in \mathcal{R}^n$, we have $0 \preceq (u+v)^+ \preceq u^+ + v^+$ and $u^- + v^- \preceq (u + v)^- \preceq 0$. Furthermore, it holds that $\|(u + v)^+\|_F \leq \|u^+\|_F + \|v^+\|_F$ and $\|(u + v)^-\|_F \leq \|u^-\|_F + \|v^-\|_F$.*

Proof. For any $u, v \in \mathcal{R}^n$, it is evident that $0 \preceq (u + v)^+ \preceq u^+ + v^+$. Hence, we have $\|(u+v)^+\|_F \leq \|u^+ + v^+\|_F \leq \|u^+\|_F + \|v^+\|_F$. Similarly, we have $u^- + v^- \preceq (u+v)^- \preceq 0$. Therefore we obtain $\|(u+v)^-\|_F \leq \|u^- + v^-\|_F \leq \|u^-\|_F + \|v^-\|_F$.

Lemma 9. *Suppose that z and $\bar{z}$ are denoted as mentioned above, then*

$$\|(\tau\bar{\mu}e - \bar{w})^+\|_F \leq \|(\Delta x \circ \Delta s)^-\|_F. \tag{15}$$

Proof. From Proposition 1, Lemma 6 and 8, we have

$$\begin{aligned}\|(\tau\bar{\mu}e - \bar{w})^+\|_F &= \|(\tau\bar{\mu}e - \bar{x} \circ \bar{s})^+\|_F = \|(t\tau\mu e - tv - \Delta x \circ \Delta s)^+\|_F \\ &\leq \|t(\tau\mu e - v)^+ + (-\Delta x \circ \Delta s)^+\|_F = \|(\Delta x \circ \Delta s)^-\|_F.\end{aligned}$$

In the rest of the paper, we choose $p = x^{1/2}$. From (13) we have $Q_p^{-1/2}\Delta x + Q_p^{1/2}\Delta s = Q_p^{-1/2}(tv - x)$. Let $h = Q_p^{-1/2}\Delta x, q = Q_p^{1/2}\Delta s, d = Q_p^{-1/2}(tv - x)$, then $h + q = d$ and we have the following result.

Lemma 10. *Suppose h, q, d are denoted as above, it holds that*

$$(i)\langle h, q \rangle = 0; \quad (ii)\|(h \circ q)^-\|_F = \|(\Delta x \circ \Delta s)^-\|_F; \quad (iii)\|(h \circ q)^-\|_F \leq \frac{1}{4}\|d\|_F^2.$$

Proof. (i) From the above definitions of h and q, we have

$$\langle h, q \rangle = \mathrm{tr}(h \circ q) = \mathrm{tr}[(Q_p^{-1/2}\Delta x) \circ (Q_p^{-1/2}\Delta s)] = \mathrm{tr}[(\Delta x) \circ \Delta s] = \langle \Delta x, \Delta s \rangle = 0.$$

(ii) Direct calculation yields

$$\|(h \circ q)^-\|_F = \|[(Q_p^{-1/2}\Delta x) \circ (Q_p^{-1/2}\Delta s)]^-\|_F = \|(\Delta x \circ \Delta s)^-\|_F.$$

(iii) By (i) and Lemma 7, taking into account $h+q = d$, we can directly obtain the desired conclusion.

Theorem 1. *Suppose that the definition of v is given in (11), and $(x, y, s) \in \mathcal{N}(\tau, \beta)$. Let $0 < \beta \leq \frac{1}{1+4r}, 1 - \frac{\alpha}{\sqrt{r}} \leq t \leq 1$, where α is the arbitrary constant satisfying $0 < \alpha \leq \sqrt{\beta\tau}$, then the new iterate point $(\bar{x}, \bar{y}, \bar{s})$ generated by the algorithm 3.1 satisfies $\|(\tau\bar{\mu}e - \bar{w})^+\|_F \leq \beta\tau\mu$.*

Proof. It follows from Proposition 2 and Lemma 10 that

$$\| (\tau\bar{\mu}e - \bar{w})^+ \|_F \leq \tfrac{1}{4}\|Q_w^{-1/2}(tv - x)\|_F^2 \leq \tfrac{1}{4}(t\|Q_p^{-1/2}(v - x)\|_F + (1 - t)\|Q_p^{-1/2}x\|_F)^2$$
$$\leq \tfrac{1}{4}(t\sqrt{\beta\tau\mu} + \tfrac{\alpha}{\sqrt{r}}\langle x, s\rangle^{\frac{1}{2}})^2 = \tfrac{1}{4}(t\sqrt{\beta\tau\mu} + \alpha\sqrt{\mu})^2 \leq \tfrac{1}{4}(t\sqrt{\beta\tau\mu} + \sqrt{\beta\tau\mu})^2 \leq \beta\tau\mu.$$

Theorem 2. *Algorithm 3.1 terminates in at most $O(\sqrt{r}L)$ iterations, where $L = \frac{1}{\alpha}\log\frac{\langle x^0, s^0\rangle}{\varepsilon}$, and $\alpha = \sqrt{\beta\tau}$.*

Proof. By Theorem 1, we have $\sigma^k \leq 1 - \frac{\alpha}{\sqrt{r}}$. It follows from Lemma 6 that

$$\mu^k \leq (1 - \frac{\alpha}{\sqrt{r}})^k\mu^0. \tag{16}$$

Therefore, if $k \geq \sqrt{r}L$, by (14) and (16), $\langle x^k, s^k\rangle \leq (1 - \frac{\alpha}{\sqrt{r}})^{\sqrt{r}L}\langle x^0, s^0\rangle \leq \varepsilon$. The theorem follows immediately from the above inequality.

Acknowledgements

The work is supported by National Natural Science Foundation of China (10571109, 10971122), Natural Science Foundation of Shandong (Y2008A01), and Scientific and Technological Project of Shandong Province (2009GG10001012).

References

1. Rangarajan, B.K.: Polynomial Convergence of Infeasible-Interior-Point Methods over Symmetric Cones. SIAM J. OPTIM. 16(4), 1211–1229 (2006)
2. Alizadeh, F., Goldfarb, D.: Second-order cone programming. Math. Program. 95, 3–51 (2003)
3. Peng, J.M., Roos, C., Terlaky, T.: Self-regular functions and new search directions for linear and semidefinite optimization. Math. Program. Ser. A 93, 129–171 (2002)
4. Toh, K.C., Tütüncü, R.H., Todd, M.J.: SDPT3 Version 3.02-A MATLAB software for semidefinite-quadratic-linear programming (2002)
5. Todd, M.J., Toh, K.C., Tütüncü, R.H.: On the Nesterov-Todd direction in semidefinite programming. SIAM Journal on Optimization 8, 769–796 (1998)
6. Shivaswamy, P.K., Bhattacharyya, C., Smola, A.J.: Second Order Cone Programming Approaches for Handling Missing and Uncertain Data. Journal of Machine Learning Research 7, 1283–1314 (2006)
7. Monteiro, R.D.C., Zhang, Y.: A unified analysis for a class of long-step primal-dual path-following interior-point algorithms for semidefinie programming. Math. Program. 81, 281–299 (1998)
8. Debnath, R., Muramatsu, M., Takahashi, H.: An Efficient Support Vector Machine Learning Method with Second-Order Cone Programming for Large-Scale Problems. Applied Intelligence 23, 219–239 (2005)
9. Nesterov, Y.E., Todd, M.J.: Primal-dual Interior-point Methods for Self-scaled Cones. SIAM J. OPTIM. 8(2), 324–364 (1998)

A One-Step Smoothing Newton Method Based on a New Class of One-Parametric Nonlinear Complementarity Functions for P_0-NCP

Liang Fang[1,⋆], Xianming Kong[1], Xiaoyan Ma[1], Han Li[2], and Wei Zhang[1]

[1] College of Mathematics and System Science, Taishan University, 271021, Tai'an, P.R. China
fangliang3@163.com
[2] Department of Mathematics, Heze University, 274015, Heze, P.R. China

Abstract. Nonlinear complementarity problem with P_0-function is studied. Based on a new class of one-parametric nonlinear complementarity functions, the problem is approximated by a family of parameterized smooth equations and a one-step smoothing Newton method is presented. The proposed algorithm only need to solve one system of linear equations and perform one line search per iteration. It is proved to be convergent globally and superlinearly without strict complementarity. Moreover, the algorithm has locally quadratic convergence under mild conditions.

Keywords: Nonlinear complementarity; Smoothing Newton method; P_0-function; Coerciveness; Global convergence.

1 Introduction

Consider the the following nonlinear complementarity problem with P_0-function (P_0-NCP(f) for simplicity): Finding a vector $x \in \mathcal{R}^n$ such that

$$x \geq 0, f(x) \geq 0, \langle x, f(x) \rangle = 0, \tag{1}$$

where $\langle \cdot, \cdot \rangle$ represents Euclidean inner product of vectors, $f : \mathcal{R}^n \to \mathcal{R}^n$ is a continuously differentiable P_0-function.

Recently, there has been strong interests in smoothing method for solving NCPs [1,4,6,2]. One motivation is that NCPs have wide applications in many fields [7,8]. The idea of smoothing Newton method is to use a smooth function to reformulate the NCP as a family of parameterized smooth equations which can be solved approximately by using Newton method. A solution of the original problem can be found by reducing the parameter to zero. However, many of existing methods depend on the assumption of strict complementarity or monotonicity at the KKT points of the problem.

Motivated by this direction, in this paper, a new class of one-parametric nonlinear complementarity functions for NCPs are investigated. Based on these

⋆ Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 110–117, 2010.

smoothing functions, we reformulate the P_0-NCP(f) as a system of nonlinear equations and propose a one-step smoothing Newton method which need to solve only one system of linear equations and perform one line search at each iteration. The algorithm can start from an arbitrary point, and the global and superlinear convergence of the method are proved without strict complementarity. Moreover, the algorithm has locally quadratic convergence if f' is Lipschitz continuous.

This paper is organized as follows. In Section 2, we recall some preliminaries, and give a new class of one-parametric nonlinear complementarity functions and their properties. We present a one-step smoothing Newton method for the P_0-NCP(f) and state some preliminary results in Section 3. The global convergence and superlinear convergence of the algorithm are investigated in Section 4.

The following notations will be used throughout this paper. $\mathcal{R}^n$ (respectively, R) denotes the space of n-dimensional real column vectors (respectively, real numbers), $\mathcal{R}^n_+$ and $\mathcal{R}^n_{++}$ denote the nonnegative and positive orthants of $\mathcal{R}^n$, R_+ (respectively, R_{++}) denotes the nonnegative (respectively, positive) orthant in R. We define $N = \{1, 2, ..., n\}$. For any $a \in \mathcal{R}$, $[a]_+ := \max\{a, 0\}$, $[a]_- :=\max\{-a, 0\}$. It is evident that $[a]_+ \geq 0, [a]_- \geq 0, a = [a]_+ - [a]_-$, and $[a]_+[a]_- = 0$. For any vector $u \in \mathcal{R}^n$, we denote by diag$\{u_i : i \in N\}$ the diagonal matrix whose ith diagonal element is u_i and vec$\{u_i : i \in N\}$ the vector u. The matrix I represents the identity matrix with suitable dimension. The symbol $\|\cdot\|$ stands for the 2-norm. For any differentiable function $f : \mathcal{R}^n \to \mathcal{R}^n$, $f'(x)$ denotes the Jacobian of f. We denote by $\Theta := \{x \in \mathcal{R}^n : x \geq 0, f(x) \geq 0, \langle x, f(x)\rangle = 0\}$ the solution set of P_0-NCP(f). For any $\alpha, \beta \in \mathcal{R}_{++}$, $\alpha = O(\beta)$ (respectively, $\alpha = o(\beta)$) means α/β is uniformly bounded (respectively, tends to zero) as $\beta \to 0$. $\mathcal{R}^n \times \mathcal{R}^m$ is identified with $\mathcal{R}^{n+m}$. For any matrix $A \in \mathcal{R}^{n\times n}$, $A \succeq 0$ ($A \succ 0$) means A is positive semi-definite (positive definite, respectively).

2 Preliminaries and a New Class of One-Parametric Nonlinear Complementarity Functions

Firstly, we recall some background materials and preliminary results.

Definition 1. *(1) A matrix $P \in \mathcal{R}^{n\times n}$ is said to be a P_0-matrix if all its principal minors are nonnegative.*

(2) A function $f : \mathcal{R}^n \to \mathcal{R}^n$ is said to be a P_0-function if for all $x, y \in \mathcal{R}^n$ with $x \neq y$, there exists an index $i_0 \in N$ such that

$$x_{i0} \neq y_{i0}, (x_{i0} - y_{i0})[f_{i0}(x) - f_{i0}(y)] \geq 0.$$

Definition 2. *Suppose that $f : \mathcal{R}^n \to \mathcal{R}^n$ is locally Lipschitz continuous around $x \in \mathcal{R}^n$. f is said to be semi-smooth at x if f is directionally differentiable at x and*

$$\lim_{V \in \partial f(x+th'), h' \to h, t \to 0^+} Vh' \quad \textit{exists for all } h \in \mathcal{R}^n,$$

where $\partial f(\cdot)$ denote the generalized derivative in the sense in [3].

Definition 3. *Let $\mathcal{K} \subset \mathcal{R}^n$ be a cone, and $f : \mathcal{K} \to \mathcal{R}^n$ is a continuous mapping. If there exists a point $u \in \mathcal{K}$ such that*

$$\lim_{\|x\| \to +\infty} \frac{(x-u)^T f(x)}{\|x\|} = +\infty, x \in \mathcal{K},$$

then the mapping f is called satisfying the coerciveness condition in $\mathcal{K}$.

For any $(a, b) \in \mathcal{R}^2$, in this paper, we consider the one-parametric functions

$$\phi(\mu, a, b) := (1 + 2\mu)(a + b) - \sqrt{(a-b)^2 + \tau ab + (4-\tau)\mu^2}, \tag{2}$$

where $\tau \in [0, 4)$ is arbitrary but fixed parameter. It is not hard to see that

(i) when $\tau = 0$, $\phi(\mu, a, b)$ reduces to the smoothing natural residual function

$$\phi_{SNR}(\mu, a, b) := (1 + 2\mu)(a + b) - \sqrt{(a-b)^2 + 4\mu^2};$$

(ii) when $\tau = 2$, $\phi(\mu, a, b)$ becomes the smoothing Fisher-Burmeister function

$$\phi_{SFB}(\mu, a, b) := (1 + 2\mu)(a + b) - \sqrt{a^2 + b^2 + 2\mu^2}.$$

Property 1. For any $(a, b) \in \mathcal{R}^2$, and $\phi(\mu, a, b)$ be defined by (2), we have

$$\phi(0, a, b) = 0 \Longleftrightarrow a \geq 0, b \geq 0, ab = 0. \tag{3}$$

Proof. If $a \geq 0, b \geq 0, ab = 0$, then $\phi(0, a, b) = \phi_{FB}(a, b) = a + b - \sqrt{a^2 + b^2}$. Since $\phi_{FB}(a, b)$ is a nonlinear complementarity function, hence $\phi(0, a, b) = 0$. On the other hand, assume $\phi(0, a, b) = a+b-\sqrt{(a-b)^2 + \tau a \circ b} = 0$. By taking inner product with $-[a]_-$, and taking into account $[a]_+ \circ [a]_- = 0$, we have

$$\begin{aligned}
0 &= \langle -[a]_-, \phi_\tau(0, a, b)\rangle \\
&= \langle -[a]_-, a + b - \sqrt{(a-b)^2 + \tau a \circ b}\rangle \\
&= \langle -[a]_-, \tfrac{4-\tau}{2}a + \tfrac{\tau-2}{2}a + b - \sqrt{(a-b)^2 + \tau a \circ b}\rangle \\
&= \langle -[a]_-, \tfrac{4-\tau}{2}a\rangle + \langle -[a]_-, \tfrac{\tau-2}{2}a + b - \sqrt{(\tfrac{\tau-2}{2}a + b)^2 + \tfrac{\tau(4-\tau)}{4}a^2}\rangle \\
&= \tfrac{4-\tau}{2}\langle -[a]_-, [a]_+ - [a]_-\rangle + \langle [a]_-, \sqrt{(\tfrac{\tau-2}{2}a + b)^2 + \tfrac{\tau(4-\tau)}{4}a^2} - (\tfrac{\tau-2}{2}a + b)\rangle \\
&= \frac{4-\tau}{2}\|[a]_-\|^2 + \langle [a]_-, \sqrt{(\frac{\tau-2}{2}a + b)^2 + \frac{\tau(4-\tau)}{4}a^2} - (\frac{\tau-2}{2}a + b)\rangle. \quad (4)
\end{aligned}$$

It follows that $\sqrt{(\frac{\tau-2}{2}a + b)^2 + \frac{\tau(4-\tau)}{4}a^2} - (\frac{\tau-2}{2}a + b) \geq 0$. Taking into account $[a]_- \geq 0$, we have $\langle [a]_-, \sqrt{(\frac{\tau-2}{2}a + b)^2 + \frac{\tau(4-\tau)}{4}a^2} - (\frac{\tau-2}{2}a + b)\rangle \geq 0$. Since $\tau \in [0, 4)$, we obtain $\frac{4-\tau}{2}\|[a]_-\|^2 \geq 0$. It follows from (4) that $\frac{4-\tau}{2}\|[a]_-\|^2 = 0$ and $\sqrt{(\frac{\tau-2}{2}a + b)^2 + \frac{\tau(4-\tau)}{4}a^2} - (\frac{\tau-2}{2}a + b) = 0$. Hence, $[a]_- = 0$. Thus $a \geq 0$. Similarly we obtain $b \geq 0$. Next we prove $ab = 0$. Since $a \geq 0, b \geq 0$ and $\phi(0, a, b) = 0$, we have $\phi_\tau(0, a, b) = a + b - \sqrt{(a-b)^2 + \tau a \circ b} = 0$. Hence $\sqrt{(a-b)^2 + \tau ab} = a + b \geq 0$. Thus $(4 - \tau)ab = 0$. Since $\tau \in [0, 4)$, we have $ab = 0$. The proof is completed.

From (2), for any $\mu \neq 0$, a straightforward calculation yields

$$\phi'_\mu(\mu, a, b) = 2(a+b) - \frac{(8-2\tau)\mu}{\sqrt{(a-b)^2 + \tau ab + (4-\tau)\mu^2}}, \tag{5}$$

$$\phi'_a(\mu, a, b) = 1 + 2\mu - \frac{a + \frac{\tau-2}{2}b}{\sqrt{(a-b)^2 + \tau ab + (4-\tau)\mu^2}}, \tag{6}$$

$$\phi'_b(\mu, a, b) = 1 + 2\mu - \frac{b + \frac{\tau-2}{2}a}{\sqrt{(a-b)^2 + \tau ab + (4-\tau)\mu^2}}. \tag{7}$$

It is evident that $\phi'_a \geq 2\mu, \phi'_b \geq 2\mu$, and $\phi'_\mu, \phi'_a, \phi'_b$ are continuous for any $\mu > 0$.

For any $z := (\mu, x) \in \mathcal{R}_+ \times \mathcal{R}^n$, let

$$G(z) := \begin{pmatrix} e^\mu - 1 \\ \Phi(z) \end{pmatrix}, \tag{8}$$

where $\Phi : \mathcal{R}_+ \times \mathcal{R}^n \to \mathcal{R}^n$ is defined by

$$\Phi(z) := \begin{pmatrix} \phi(\mu, x_1, f_1(x)) \\ \vdots \\ \phi(\mu, x_n, f_n(x)) \end{pmatrix}. \tag{9}$$

It is obviously that Φ is continuously differentiable at any $z = (\mu, x) \in \mathcal{R}_{++} \times \mathcal{R}^n$.

Define merit function $\Psi : \mathcal{R}_+ \times \mathcal{R}^n \to \mathcal{R}_+$ by

$$\Psi(z) := \|G(z)\|^2 = (e^\mu - 1)^2 + \|\Phi(z)\|^2. \tag{10}$$

From (3), we know that the P_0-NCP(f) is equivalent to the equation $G(z) = 0$.

Theorem 1. *Let $z := (\mu, x) \in \mathcal{R}_+ \times \mathcal{R}^n$ and $G(z)$ be defined by (8) and (9), then the following results hold.*

(i) $G(z)$ is continuously differentiable at any $z \in \mathcal{R}_{++} \times \mathcal{R}^n$ with its Jacobian

$$G'(z) = \begin{pmatrix} e^\mu & 0 \\ B(z) & C(z) \end{pmatrix}, \tag{11}$$

where

$$B(z) := vec\left\{ 2(x_i + f_i(x)) - \frac{(8-2\tau)\mu}{\sqrt{(x_i - f_i(x))^2 + \tau x_i f_i(x) + (4-\tau)\mu^2}} : i \in N \right\},$$

$$C(z) := C_1(z) + C_2(z) f'(z),$$

$$C_1(z) := (1+2\mu)I - diag\left\{ \frac{x_i + \frac{\tau-2}{2} f_i(x)}{\sqrt{(x_i - f_i(x))^2 + \tau x_i f_i(x) + (4-\tau)\mu^2}} : i \in N \right\},$$

$$C_2(z) := (1+2\mu)I - diag\left\{ \frac{f_i(x) + \frac{\tau-2}{2} x_i}{\sqrt{(x_i - f_i(x))^2 + \tau x_i f_i(x) + (4-\tau)\mu^2}} : i \in N \right\}.$$

(ii) If f is a P_0 function, $G'(z)$ is nonsingular for any $z \in \mathcal{R}_{++} \times \mathcal{R}^n$.

Proof. (i) Note that $\Phi(\mu, x)$ is continuously differentiable at any $z \in \mathcal{R}_{++} \times \mathcal{R}^n$. So $G(z)$ defined by (8) is also continuously differentiable at any $z \in \mathcal{R}_{++} \times \mathcal{R}^n$. For any $\mu > 0$, a direct calculation from (8) yields (11).

Next we prove (ii). By (6) and (7), we obtain $C_1(z) \succ 0$ and $C_2(z) \succ 0$. Since f is a P_0-function, by Theorem 2.8 in [5], $f'(x)$ is a P_0-matrix for all $x \in \mathcal{R}^n$. Taking into account the fact that $C_2(z)$ is a positive diagonal matrix, it follows from a straightforward calculation that all principal minors of $C_2(z)f'(z)$ are non-negative. By Definition 1, $C_2(z)f'(z)$ is a P_0-matrix. Hence, by Theorem 3.3 in [1], $C_1(z)+C_2(z)f'(z)$ is invertible. Thus, the matrix $G'(z)$ is non-singular.

3 Description of the Algorithm

The aim of this section is to propose the new one-step smoothing Newton method for the P_0-NCP(f) and show its well-definedness.

Algorithm 3.1 (A new one-step smoothing Newton method for P_0-NCP(f)).

Step 0. Choose constants $\delta \in (0,1), \sigma \in (0,1)$, and an arbitrary initial point $z^0 := (\mu_0, x^0) \in \mathcal{R}_{++} \times \mathcal{R}^n$. Let $\eta = \sqrt{\Psi(z^0)} + 1$ and $\bar{\mu} = \mu_0, \bar{z} := (\bar{\mu}, 0) \in \mathcal{R}_{++} \times \mathcal{R}^n$. Choose $\gamma \in (0,1)$ such that

$$\gamma \bar{\mu} \eta < \frac{1}{2}. \tag{12}$$

Set $k := 0$.

Step 1. If $\Psi(z^k) = 0$, then stop. Else, let

$$\beta_k := \beta(z^k) = e^{\mu_k} \gamma \min\{1, \Psi(z^k)\}. \tag{13}$$

Step 2. Compute $\Delta z^k := (\Delta\mu_k, \Delta x^k) \in \mathcal{R} \times \mathcal{R}^n$ by

$$G(z^k) + G'(z^k)\Delta z^k = \beta_k \bar{z}. \tag{14}$$

Step 3. Let $\lambda_k = \max\{\delta^l | l = 0, 1, 2, \ldots\}$ such that

$$\Psi(z^k + \delta^l \Delta z^k) \leq \left[1 - \sigma(1 - 2\gamma\eta\bar{\mu})\delta^l\right] \Psi(z^k). \tag{15}$$

Step 4. Set $z^{k+1} := z^k + \lambda_k \Delta z^k$ and $k := k + 1$. Go to step 1.

Note that if $\Psi(z^k) = 0$, then (x^k, y^k) is the solution of the P_0-NCP(f). So, the stopping criterion in Step 1 is reasonable. Define the set

$$\Omega := \{z = (\mu, x) \in \mathcal{R}_+ \times \mathcal{R}^n | \mu \geq \gamma \min\{1, \Psi(z)\}\bar{\mu}\}.$$

Lemma 1 (Lemma 4.2, [4]). *For any $\mu \geq 0$,*

$$-\mu \leq \frac{1 - e^{\mu}}{e^{\mu}} \leq -\mu e^{-\mu}. \tag{16}$$

Theorem 2. *Algorithm 3.1 is well defined and generates an infinite sequence $\{z^k := (\mu_k, x^k)\}$ with $\mu_k > 0$ and $z^k \in \Omega$ for all $k \geq 0$.*

Proof. If $\mu_k > 0$, since f is a continuously differentiable P_0-function, by Theorem 1, $G'(z^k)$ is non-singular. Hence, Step 2 is well-defined at the kth iteration.

For any $\alpha \in (0,1]$, from (14) we have

$$\Delta\mu_k = \frac{1-e^{\mu_k}}{e^{\mu_k}} + \frac{\beta_k\bar{\mu}}{e^{\mu_k}}. \tag{17}$$

It follows from Lemma 1 and (17) that for any $\alpha \in (0,1]$,

$$\mu_{k+1} = \mu_k + \alpha\Delta\mu_k = \mu_k + \alpha\left(\frac{1-e^{\mu_k}}{e^{\mu_k}} + \frac{\beta_k\bar{\mu}}{e^{\mu_k}}\right) \geq (1-\alpha)\mu_k + \alpha\gamma\bar{\mu}\min\{1,\Psi(z^k)\} > 0.$$

By Taylor expansion and (17), we have

$$\begin{aligned} e^{\mu_k+\alpha\Delta\mu_k} - 1 &= e^{\mu_k}[1+\alpha\Delta\mu_k + O(\alpha^2)] - 1 \\ &= (1-\alpha)(e^{\mu_k}-1) + \alpha\beta_k\bar{\mu} + O(\alpha^2). \end{aligned} \tag{18}$$

By $\beta_k^2 = e^{2\mu_k}\gamma^2(\min\{1,\Psi(z^k)\})^2 \leq e^{2\mu_k}\gamma^2\Psi(z^k)$, $\beta_k \leq e^{\mu_k}\gamma\sqrt{\Psi(z^k)}$. It follows from (8) and (13) that $e^{\mu_k} - 1 \leq \sqrt{\Psi(z^k)}$, and $e^{\mu_k} \leq \eta$. So, we have

$$\begin{aligned} (e^{\mu_k+\alpha\Delta\mu_k}-1)^2 &= (1-\alpha)^2(e^{\mu_k}-1)^2 + 2\alpha(1-\alpha)\beta_k(e^{\mu_k}-1)\bar{\mu} + \alpha^2\beta_k^2\bar{\mu}^2 + O(\alpha^2) \\ &\leq (1-\alpha)(e^{\mu_k}-1)^2 + 2\alpha\gamma\sqrt{\Psi(z^k)}e^{\mu_k}(e^{\mu_k}-1)\bar{\mu} + O(\alpha^2) \\ &\leq (1-\alpha)(e^{\mu_k}-1)^2 + 2\alpha\gamma\eta\Psi(z^k)\bar{\mu} + O(\alpha^2). \end{aligned} \tag{19}$$

On the other hand, it follows from (14) that $\Phi(z^k) + \Phi'(z^k)\Delta z^k = 0$. Hence

$$\begin{aligned} \|\Phi(z^k+\alpha\Delta z^k)\|^2 &= \|\Phi(z^k) + \alpha\Phi'(z^k)\Delta z^k + o(\alpha)\|^2 \\ &= \|(1-\alpha)\Phi(z^k) + o(\alpha)\|^2 \\ &\leq (1-\alpha)\|\Phi(z^k)\|^2 + o(\alpha). \end{aligned} \tag{20}$$

From (8), (19) and (20), we have

$$\begin{aligned} \Psi(z^k+\alpha\Delta z^k) &= (e^{\mu_k+\alpha\Delta\mu_k}-1)^2 + \|\Phi(z^k+\alpha\Delta z^k)\|^2 \\ &\leq (1-\alpha)(e^{\mu_k}-1) + \alpha\beta_k\bar{\mu} + (1-\alpha)\|\Phi(z^k)\|^2 + o(\alpha) \\ &\leq (1-\alpha)(e^{\mu_k}-1)^2 + 2\alpha\gamma\eta\Psi(z^k)\bar{\mu} + (1-\alpha)\|\phi(z^k)\|^2 + o(\alpha) \\ &\leq (1-\alpha)\Psi(z^k) + 2\alpha\gamma\eta\Psi(z^k)\bar{\mu} + o(\alpha) \\ &= [1-(1-2\gamma\eta\bar{\mu})\alpha]\Psi(z^k) + o(\alpha). \end{aligned}$$

Since $\gamma\eta\bar{\mu} < 1/2$, there exists $\bar{\alpha} \in (0,1]$, such that $\alpha \in (0,\bar{\alpha}]$, and $\Psi(z^k+\Delta z^k) \leq [1-\sigma(1-2\gamma\eta\bar{\mu})\alpha]\Psi(z^k)$. Thus, Step 3 is well defined. Therefore Algorithm 3.1 is well defined and generates an infinite sequence $\{z^k\}$ with $\mu_k > 0$.

Next, we prove $z^k \in \Omega$ by induction on k. Obviously, $\mu_0 \geq \gamma\min\{1,\Psi(z^k)\}\bar{\mu}$. Suppose that $z^k \in \Omega$, i.e., $\mu_k \geq \gamma\min\{1,\Psi(z^k)\}\bar{\mu}$, then by (15)-(17), we have

$$\mu_{k+1} = \mu_k + \alpha\Delta\mu_k = \mu_k + \alpha\left(\frac{1-e^{\mu_k}}{e^{\mu_k}} + \frac{\beta_k\bar{\mu}}{e^{\mu_k}}\right)$$

$$
\begin{aligned}
&\geq \mu_k + \alpha\left(-\mu_k + \frac{\gamma e^{\mu_k}\min\{1, \Psi(z^k)\}\bar{\mu}}{e^{\mu_k}}\right)\\
&\geq (1-\alpha)\mu_k + \alpha\gamma\bar{\mu}\min\{1, \Psi(z^k)\},\\
&\geq (1-\alpha)\gamma\bar{\mu}\min\{1, \Psi(z^k)\} + \alpha\gamma\bar{\mu}\min\{1, \Psi(z^k)\}\\
&= \gamma\bar{\mu}\min\{1, \Psi(z^k)\} \geq \gamma\bar{\mu}\min\{1, \Psi(z^{k+1})\}.
\end{aligned}
$$

4 Convergence Analysis

In this section, we analyze the convergence properties of Algorithm 3.1.

Lemma 2. *Let $\Phi(\mu, x)$ be defined by (9). For any $\mu, c > 0$, define the level set*

$$L_\mu(c) := \{x \in \mathcal{R}^n : \|\Phi(\mu, x)\| \leq c\}. \tag{21}$$

Then, for any $\mu_2 \geq \mu_1 > 0$, $c > 0$, the set $L(c) := \bigcup_{\mu_1 \leq \mu \leq \mu_2} L_\mu(c)$ is bounded.

From Lemma 2, we know that the set $L_\mu(c) := \{x \in \mathcal{R}^n : \|\Phi(\mu, x)\| \leq c\}$ is bounded for any $\mu > 0$. We can immediately get the following result.

Corollary 1. *Suppose that f is a P_0-function and $\mu > 0$. Then the function $\|\Phi(\mu, x)\|^2$ is coercive, i.e., $\lim_{\|x\|\to+\infty} \|\Phi(\mu, x)\|^2 = +\infty$.*

Lemma 3. *Let $\Psi(\cdot)$ be defined by (8) and $\{z^k := (\mu_k, x^k)\}$ be the iteration sequence generated by Algorithm 3.1. Then the sequence $\{\Psi(z^k)\}$ is convergent. If it does not converge to zero, then $\{z^k := (\mu_k, x^k)\}$ is bounded.*

Proof. It followed from Step 3 and Theorem 2 that $\{\Psi(z^k)\}$ is monotonically decreasing and $\{z^k\} \in \Omega$. We know that $\{\Psi(z^k)\}$ is convergent. Then there exists Ψ^* such that $\Psi(z^k) \to \Psi^*$ as $k \to +\infty$. If $\{\Psi(z^k)\}$ does not converge to zero, then $\Psi^* > 0$. By $\{z^k\} \subset \Omega$ and $\mu_k \leq e^{\mu_k} - 1 \leq f(z^k) \leq f(z^0)$, $\{\mu_k\}$ is bounded. Obviously, there exist $\mu_1, \mu_2 > 0$ such that $0 < \mu_1 \leq \mu_k \leq \mu_2$ for all $k \geq 0$. Let $c_0 := \|\Psi(z^0)\|$ and $L(c_0) := \bigcup_{\mu_1 \leq \mu_k \leq \mu_2} L_{\mu_k}(c_0)$, where $L_{\mu_k}(c_0)$ is defined by (21). By $x^k \in L_{\mu_k}(c_0)$, we have $x^k \in L(c_0)$. It follows from Lemma 2 that $L(c_0)$ is bounded and hence $\{x^k\}$ is bounded. Therefore, $\{z^k\}$ is bounded.

Theorem 3. *(*Global convergence*) Suppose that f is a continuously differentiable P_0-function, the sequence $\{z^k = (\mu_k, x^k)\}$ is generated by Algorithm 3.1, and the solution set Θ of P_0-SOCCP (1) is non-empty and bounded, then $\{z^k\}$ has at least one accumulation point $\{z^* = (\mu_*, x^*)\}$ with $x^* \in \Theta$, and any accumulation point of $\{z^k\}$ is a solution of $G(z) = 0$.*

Proof. From Corollary 1, the smoothing functions defined by (2), and $G(z)$ defined by (8) have coerciveness. So, the level set $L(c)$ is bounded and the infinite sequence $\{z^k\}$ generated by Algorithm 3.1 has at least one accumulation point. Without loss of generality, we assume that $z^* = (\mu_*, x^*)$ is the limit point of the sequence $\{z^k\}$ as $k \to \infty$. It follows from the continuity of $G(\cdot)$ that $\|G(z^k)\|$ converges to a non-negative number $\|G(z^*)\|$. From the definition of $\beta(\cdot)$, we obtain that β_k is monotonically decreasing, and converges to $\beta_* = e^{\mu_*}\gamma\min\{1, \Psi(z^*)\}$.

Now, we prove $G(z^*) = 0$ by contradiction. In fact, if $G(z^*) \neq 0$, then $\|G(z^*)\| > 0$. For $\mu_k \in \Omega$, we have $0 < \beta_* \mu_0 \leq \mu_*$. By Theorem 1, there exists a closed neighborhood $\mathcal{N}(z^*)$ of z such that for any $z \in \mathcal{N}(z^*)$, we have $\mu \in \mathcal{R}_{++}$ and $G'(z)$ is invertible. Then, for any $z \in \mathcal{N}(z^*)$, let $\Delta z := (\Delta\mu, \Delta x) \in \mathcal{R} \times \mathcal{R}^n$ be the unique solution of $G(z) + G'(z)\Delta z = \beta(z)\bar{z}$. By following the proof of Lemma 5 in [6] we can find $\bar{\alpha} \in (0, 1]$ such that $\Psi(z + \alpha\Delta z) \leq [1 - \sigma(1 - 2\gamma\eta\bar{\mu})\alpha]\,\Psi(z)$ for any $\alpha \in (0, \bar{\alpha}]$, $z \in \mathcal{N}(z^*)$. Therefore, for a nonnegative integer l such that $\delta^l \in (0, \bar{\alpha}]$, we have for all sufficiently large k, $l^k \leq l$. Since $\delta^{l^k} \geq \delta^l$, it follows from (15) that $\Psi(z^{k+1}) \leq [1 - \sigma(1 - 2\gamma\eta\bar{\mu})\delta^{l^k}]\Psi(z^k) \leq [1 - \sigma(1 - 2\gamma\eta\bar{\mu})\delta^l]\Psi(z^k)$. This contradicts the fact that $\{\Psi(z^k)\}$ converges to $\Psi(z^*) = \|G(z^*)\|^2 > 0$.

Next we give the local convergence of Algorithm 3.1.

Theorem 4. *(*Local Convergence*) Suppose that f is a continuously differentiable P_0-function and z^* is an accumulation point of the iteration sequence $\{z^k\}$ generated by Algorithm 3.1. If all $V \in \partial G(z^*)$ are nonsingular, then*

(i) $\lambda_k \equiv 1$ for all z^k sufficiently close to z^*.

(ii) The whole sequence $\{z^k\}$ superlinearly converges to z^*, i.e., $\|z^{k+1} - z^*\| = o(\|z^k - z^*\|)$, and $\mu_{k+1} = o(\mu_k)$. Furthermore, if f' is Lipschitz continuous on $\mathcal{R}^n$, then $\|z^{k+1} - z^*\| = O(\|z^k - z^*\|^2)$, and $\mu_{k+1} = O(\mu_k^2)$.

Proof. The proof is similar to that of Theorem 8 in [6]. We omit it here.

References

1. Chen, B., Harker, P.T.: Smoothing approximations to nonlinear complementarity problems. SIAM Journal on Optimization 7(1), 403–420 (1997)
2. Ma, C.-F., Chen, X.: The convergence of a one-step smoothing Newton method for P_0-NCP based on a new smoothing NCP-function. Journal of Computational and Applied Mathematics 216(1), 1–13 (2008)
3. Clark, F.H.: Optimization and Non-smooth Analysis. JohnWiley and Sons, New York (1993)
4. Jiang, H.: Smoothed Fischer-Burmeister equation methods for the complementarity problem. Technical Report, Department of Mathematics, The University of Melbourne, Parville, Victoria, Australia (1997)
5. Moré, J.J., Rheinboldt, W.C.: On P- and S-functions and related classes of n-dimensional non-linear mappings. Linear Algebra and Applications 6(1), 45–68 (1973)
6. Qi, L., Sun, D., Zhou, G.: A new look at smoothing Newton methods for nonlinear complementarity problems and box constrained variational inequalities. Math. Program. Ser. A 87, 1–35 (2000)
7. Ferris, M.C., Pang, J.-S.: Engineering and economic applications of complementarity problems. SIAM Review 39(3), 669–713 (1997)
8. Harker, P.T., Pang, J.-S.: Finite-dimensional variational inequality and non-linear complementarity problems: A survey of theory, algorithms and applications. Math. Program. 48(1), 161–220 (1990)

A Neural Network Algorithm for Solving Quadratic Programming Based on Fibonacci Method

Jingli Yang and Tingsong Du

Institute of Nonlinear and Complex System, China Three Gorges University, YiChang, Hubei 443002, China
jennyyang_2008@163.com

Abstract. In this paper, a novel neural network algorithm is proposed, which solve the quadratic programming problem with linear constraints based on Fibonacci method. Compared with the existing models for solving the quadratic programming problem with linear constraints, it is more universal, since the objective function of the quadratic programming not only can be convex function but also can be quasi convex function. Finally, example is provided to show the applicability of the proposed neural network algorithm.

Keywords: Quadratic programming, Fibonacci method, Neural network, Learning algorithm.

1 Introduction

Optimization problems arise in a wide variety of scientific and engineering applications, including regression analysis, signal processing, system identification, filter design, robot control, function approximation, *etc*[1-2]. Many engineering problems can be solved by transforming the original problems into linearly constrained quadratic programming.

Compared with traditional numerical methods, the neural network approach can solve the optimization problems much faster in running times. Therefore, neural network methods for optimization problems have been received considerable attention. In 1985, Hopfield and Tank first proposed a neural network for solving Traveling Salesman Problem(TSP)[3]. Kennedy and Chua proposed a modified model and canonical circuit models that are superior to the Hopfield and Tank model. By using penalty parameter, they proposed a neural network for solving nonlinear programming problems[4]. Lately, many researchers successively proposed a number of models. In [5], Bouzerdoum and Pattison presented a neural network for solving convex quadratic optimization problems with bounded constraints. Liang &Wang and Xia &Wang presented several neural networks for solving nonlinear convex optimization with bounded constrains and box constraints, respectively [6-9]. He D.X.[10] presented a neural network for solving linear program based on bisection method. Nevertheless, majority of

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 118–125, 2010.

these methods are virtue of energy function, used back propagation(BP) algorithm. While BP algorithm is one order algorithm, it's slowly and usually reach the local solution.

In this paper, by employing Fibonacci method and the characteristic of common network topology, we present a neural network algorithm for solving the quadratic programming with linear constraints. The objective function of the quadratic programming is broad, which can be convex function or quasi convex function.

The paper is organized as follows. In Section 2, we introduce the theoretical foundation about Fibonacci method and quadratic programming. In Section 3, we present the neural network based on Fibonacci method. The neural network algorithm is proposed in Section 4. In Section 5, a example is discussed to evaluate the effectiveness of the proposed neural network algorithm. Finally, the conclusions are made in Section 6.

2 Preliminaries

2.1 Quadratic Programming Problem

We consider the following quadratic programming problem:

$$\begin{cases} \text{minimize} & f(x) = \frac{1}{2}x^T Ax + c^T x \\ \text{subject to} & l \leq Dx \leq h \end{cases} \tag{1}$$

where x=$(x_1, x_2, \cdots, x_n)^T \in R^n$, $A \in R^{n\times n}$, $D \in R^{n\times n}$is a nonsingular positive matrix, $c \in R^n$,and $l, h \in R^n$.

Remark. Clearly, when $D = I$, the problem (1) is quadratic programming problem with bound constraints[9].

2.2 Description of the Fibonacci Method

The Fibonacci numbers are defined as follows:

$$\begin{cases} F_0 = F_1 = 1, \\ F_{k+1} = F_k + F_{k-1}, k = 1, 2, \ldots \end{cases} \tag{2}$$

the formula can be showed as

$$F_k = \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^{k+1} - (\frac{1-\sqrt{5}}{2})^{k+1}], k = 0, 1, 2, \ldots \tag{3}$$

The basic idea for Fibonacci method to solve optimization problem is when we know the first search interval [a_1,b_1] and the iteration precision ε, we can obtain the last search interval [a_n,b_n] after iterating n times. We can find that

$$b_n - a_n \leq \varepsilon.$$

The shortening rate for the search interval is that

$$b_{k+1} - a_{k+1} \leq \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k).$$

Since

$$b_n - a_n = \frac{F_1}{F_2}(b_{n-1} - a_{n-1}) = \frac{F_1}{F_2} \cdot \frac{F_2}{F_3} \cdots \frac{F_{n-1}}{F_n}(b_1 - a_1) = \frac{1}{F_n}(b_1 - a_1).$$

Then

$$\frac{1}{F_n}(b_1 - a_1) \leq \varepsilon,$$

thus

$$F_n \geq \frac{b_1 - a_1}{\varepsilon}. \tag{4}$$

According to (4), we can get the minimum F_n. By using (3), we can obtain n, then the search times n can be found. By using the procedure below we can find the optimal solution for optimization problems:

Step 1: Give the constrain condition $a_1 \leq x \leq b_1$ and the iteration precision ε, according to (3)and (4)we know n, then we can get F_n, F_{n-1}, F_{n-2}, set

$$\lambda_1 = a_1 + (1 - \frac{F_{n-1}}{F_n})(b_1 - a_1) = \frac{F_{n-1}}{F_n}a_1 + (1 - \frac{F_{n-1}}{F_n})b_1 = \frac{F_{n-1}}{F_n}a_1 + \frac{F_{n-2}}{F_n}b_1,$$

$$\mu_1 = a_1 + \frac{F_{n-1}}{F_n}(b_1 - a_1) = (1 - \frac{F_{n-1}}{F_n})a_1 + \frac{F_{n-1}}{F_n}b_1 = \frac{F_{n-2}}{F_n}a_1 + \frac{F_{n-1}}{F_n}b_1,$$

put $k = 1$.

Step 2: If $|b_k - a_k| < \varepsilon$, end. The optimal solution $x^* \in [a_k, b_k]$, let $x^* = (a_k + b_k)/2$, otherwise, let

$$\begin{aligned}\lambda_k = a_k + (1 - \frac{F_{n-k}}{F_{n-k+1}})(b_k - a_k) &= \frac{F_{n-k}}{F_{n-k+1}}a_k + (1 - \frac{F_{n-k}}{F_{n-k+1}})b_k \\ &= \frac{F_{n-k}}{F_{n-k+1}}a_k + \frac{F_{n-k-1}}{F_{n-k+1}}b_k,\end{aligned}$$

$$\begin{aligned}\mu_k = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k) &= (1 - \frac{F_{n-k}}{F_{n-k+1}})a_k + \frac{F_{n-k}}{F_{n-k+1}}b_k \\ &= \frac{F_{n-k-1}}{F_{n-k+1}}a_k + \frac{F_{n-k}}{F_{n-k+1}}b_k,\end{aligned}$$

then we can get $f(\lambda_k)$, $f(\mu_k)$. If $f(\lambda_k) > f(\mu_k)$, turn to *step 3*, otherwise, turn to *step 4*.

Step 3: Let $a_{k+1} = \lambda_k$, $b_{k+1} = b_k$, and let

$$\lambda_{k+1} = \mu_k,$$

$$\mu_{k+1} = a_{k+1} + \frac{F_{n-k}}{F_{n-k+1}}(b_{k+1} - a_{k+1}) = \frac{F_{n-k-1}}{F_{n-k+1}}a_{k+1} + \frac{F_{n-k}}{F_{n-k+1}}b_{k+1},$$

calculate $f(\mu_{k+1})$.

Step 4: Let $a_{k+1} = a_k$, $b_{k+1} = \mu_k$, and let

$$\lambda_{k+1} = a_{k+1} + (1 - \frac{F_{n-k}}{F_{n-k+1}})(b_{k+1} - a_{k+1}) = \frac{F_{n-k}}{F_{n-k+1}} a_{k+1} + \frac{F_{n-k-1}}{F_{n-k+1}} b_{k+1},$$

$$\mu_{k+1} = \lambda_k,$$

calculate $f(\lambda_{k+1})$.

Step 5: Let $k = k + 1$, turn to *Step 2*.

3 Neural Network Structure

According to Fibonacci method, we design the neural network as follow:

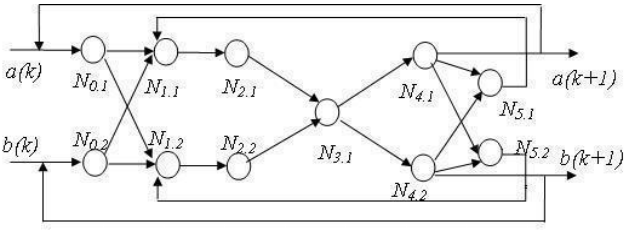


Fig. 1. Neural network structure based on Fibonacci method

We suppose that:$N_{i,j}$ means the neuron j in layer i, $net_{i,j}$ denote as the input value of neuron $N_{i,j}$, $O_{i,j}$ means the output value of neuron $N_{i,j}$, and $\omega_{(i,j),(k,l)}$ means the connection weight from $N_{i,j}$ to $N_{k,l}$. Biasing threshold vector is defined by $\theta = 0$.

The neural network structure in Fig.1 contains 6 layers, including input layer, three hidden layers, one feedback layer and output layer. We arrange calculation steps below to solve quadratic programming.

(I) Input Layer: Let $[a_k, b_k]$ as the input of $N_{0,1}$ and $N_{0,2}$, then

$$net_{0,1} = a_k = a_1,$$
$$net_{0,2} = b_k = b_1,$$

the activation functions are defined by $\varphi_{0,1}(x) = x$, $\varphi_{0,2}(y) = y$, then the outputs of $N_{0,1}$, $N_{0,2}$ are

$$O_{0,1} = \varphi_{0,1}(net_{0,1}) = a_1,$$
$$O_{0,2} = \varphi_{0,1}(net_{0,2}) = b_1.$$

(II) Hidden Layer: For the first hidden layer,$N_{1,1}$ enforces λ_k , $N_{1,2}$ enforces μ_k, the corresponding connection weights are

$$\omega_{(0,1),(1,1)} = \omega_{(0,2),(1,2)} = \frac{F_{n-k}}{F_{n-k+1}},$$

$$\omega_{(0,2),(1,1)} = \omega_{(0,1),(1,2)} = 1 - \frac{F_{n-k}}{F_{n-k+1}} = \frac{F_{n-k-1}}{F_{n-k+1}},$$

then the output of neurons $N_{1,1}$ and $N_{1,2}$ are

$$O_{1,1} = \omega_{(0,1),(1,1)} * a_k + \omega_{(0,2),(1,1)} * b_k = \lambda_k,$$
$$O_{1,2} = \omega_{(0,1),(1,2)} * a_k + \omega_{(0,2),(1,2)} * b_k = \mu_k.$$

For the second hidden layer, $N_{2,1}$ and $N_{2,2}$ are used to enforce the output of λ_k, μ_k, respectively. Set $\omega_{(1,1),(2,1)} = \omega_{(1,2),(2,2)} = 1$, then

$$O_{2,1} = f(\omega_{(1,1),(2,1)} * \lambda_k) = f(\lambda_k),$$
$$O_{2,2} = f(\omega_{(1,2),(2,2)} * \mu_k) = f(\mu_k).$$

For the third hidden layer, the input is

$$net_{3,1} = (\omega_{(2,1),(3,1)} * O_{2,1}) - (\omega_{(2,2),(3,1)} * O_{2,2}),$$

let $\omega_{(2,1),(3,1)} = \omega_{(2,2),(3,1)} = 1$, setting the activation function is

$$\varphi(net_{3,1}) = \begin{cases} 1, & \text{if } net_{3,1} > 0, \\ 0, & \text{otherwise,} \end{cases}$$

then the output is

$$O_{3,1} = f(\lambda_k) - f(\mu_k).$$

(III) Output Layer: There are two neurons in the output layer,which are used to calculate a_{k+1}, b_{k+1}, set

$$\omega_{(3,1),(4,1)} = \omega_{(3,1),(4,2)} = 1,$$

then the outputs are

$$O_{4,1} = O_{3,1} * O_{1,1} + (1 - O_{3,1}) * O_{0,1} = \begin{cases} a_{k+1} = \lambda_k = O_{1,1}, & O_{3,1} = 1, \\ a_{k+1} = a_k = O_{0,1}, & O_{3,1} = 0, \end{cases}$$

$$O_{4,2} = O_{3,1} * O_{0,2} + (1 - O_{3,1}) * O_{1,2} = \begin{cases} b_{k+1} = b_k = O_{0,2}, & O_{3,1} = 1, \\ b_{k+1} = \mu_k = O_{1,2}, & O_{3,1} = 0. \end{cases}$$

(IV) Feedback Layer: The neurons $N_{5,1}$ and $N_{5,2}$ in feedback layer are used to calculate λ_{k+1} and μ_{k+1},

$$O_{5,1} = \frac{F_{n-k}}{F_{n-k+1}}(1 - O_{3,1}) * O_{4,1} + \frac{F_{n-k-1}}{F_{n-k+1}}(1 - O_{3,1}) * O_{4,2} + O_{3,1} * O_{1,2}$$
$$= \begin{cases} \lambda_{k+1} = O_{1,2}, & O_{3,1} = 1, \\ \lambda_{k+1} = \frac{F_{n-k}}{F_{n-k+1}} O_{4,1} + \frac{F_{n-k-1}}{F_{n-k+1}} O_{4,2}, & O_{3,1} = 0, \end{cases}$$

$$O_{5,2} = \frac{F_{n-k-1}}{F_{n-k+1}} O_{3,1} * O_{4,1} + \frac{F_{n-k}}{F_{n-k+1}} O_{3,1} * O_{4,2} + (1 - O_{3,1}) * O_{1,1}$$
$$= \begin{cases} \mu_{k+1} = \frac{F_{n-k-1}}{F_{n-k+1}} O_{4,1} + \frac{F_{n-k}}{F_{n-k+1}} O_{4,2}, & O_{3,1} = 1, \\ \mu_{k+1} = O_{1,1}, & O_{3,1} = 0. \end{cases}$$

(V) Iteration: Let the output neurons as the input of the feedback layer, If $f(\lambda_k) > f(\mu_k)$, then $a_{k+1} = \lambda_k$, $b_{k+1} = b_k$, otherwise, let $a_{k+1} = a_k$, $b_{k+1} = \mu_k$.

Let $k = 1$, $k = k + 1$,circulation,until $|b_k - a_k| \leq \varepsilon$.

4 The Neural Network Algorithm Based on Fibonacci Method

For the quadratic programming subject to linear constraints, we can present the algorithm based on the neural network structure in Section 3.

Step 1: According to the constrains condition $l \leq Dx \leq h$, we can obtain the upper bound and lower bound of x, denote a_1, b_1. Let $net_{0,1} = a_1, net_{0,2} = b_1$, give $\varepsilon \geq 0$ as iteration precision. If $|b_1 - a_1| < \varepsilon$, let optimal solution $x^* = (a_1 + b_1)/2$, otherwise, let the initial solution is $x(1) = (a_1 + b_1)/2$, turn to *Step 2*.

Step 2: Calculate $O_{1,1} = \lambda_1$, $O_{1,2} = \mu_1$, $O_{2,1} = f(\lambda_1)$, $O_{2,2} = f(\mu_1)$, let $O_{3,1} = f_(\lambda_1) - f(\mu_1)$.

Step 3: If $O_{3,1} > 0$, let $O_{4,1} = a_2 = \lambda_1$,$O_{4,2} = b_2 = b_1$, otherwise, let $O_{4,1} = a_2 = a_1$, $O_{4,2} = b_2 = \mu_1$.

Step 4: When $O_{3,1} > 0$, let

$$O_{5,1} = \lambda_2 = \mu_1,$$
$$O_{5,2} = \mu_2 = a_2 + \frac{F_{n-2}}{F_{n-2+1}}(b_2 - a_2) = \frac{F_{n-2-1}}{F_{n-2+1}}a_2 + \frac{F_{n-2}}{F_{n-2+1}}b_2.$$

When $O_{3,1} \leq 0$, let

$$O_{5,1} = \lambda_2 = a_2 + (1 - \frac{F_{n-2}}{F_{n-2+1}})(b_2 - a_2) = \frac{F_{n-2}}{F_{n-2+1}}a_2 + \frac{F_{n-2-1}}{F_{n-2+1}}b_2,$$
$$O_{5,2} = \mu_2 = \lambda_1.$$

Step 5: Let $k = 1$, $k = k+1$, and let $net_{0,1} = a_k$, $net_{0,2} = b_k$, until $|b_k - a_k| \leq \varepsilon$, then we can obtain $x^* = (a_k + b_k)/2$.

Theorem. Used the neural network algorithm based on Fibonacci method to solve quadratic programming problem, the error of the approximate solution is less than $\tau^k|b_1 - a_1|$, where $\tau = \frac{F_{n-k}}{F_{n-k+1}}$. And we can conclude that

$$\lim_{k\to\infty} \frac{F_{k-1}}{F_k} = \frac{\sqrt{5}-1}{2} = 0.618. \tag{5}$$

Proof. According to the neural network algorithm we proposed, we can get $[a_k, b_k]$,$k = 1, 2, \cdots, n$, while $|b_k - a_k| = \tau^k|b_1 - a_1| \to 0$,$(k \to \infty)$, then when $|b_k - a_k| < \varepsilon$, let $x^* = \frac{a_k+b_k}{2}$ as the approximate optimal solution, then the error is less then $\tau^k|b_1 - a_1|$, where $\tau = \frac{F_{n-k}}{F_{n-k+1}}$.

According to (2), we can easily obtain (5).

5 Simulation Result

In this section, we give an illustrative example. The simulation is conducted in MATLAB.

Consider a quadratic programming example as

$$\begin{cases} \min & f(x) = x_1^2 + 4x_2^2 - 4x_1x_2 - 2x_1 \\ \text{s. t.} & 1 \le -x_1 + x_2 \le 2 \\ & 2 \le 2x_1 + 3x_2 \le 3. \end{cases} \tag{6}$$

Then

$$A = \begin{pmatrix} 2 & -4 \\ -4 & 8 \end{pmatrix}, \quad c = \begin{pmatrix} -2 \\ 0 \end{pmatrix},$$

and

$$D = \begin{pmatrix} -1 & 1 \\ 2 & 3 \end{pmatrix}, \quad l = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad h = \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

This problem has a unique optimal solution $x^* = (-0.2, 0.8)^T$ and the optimal value $f^* = 3.64$ when solved by the neural network model in [9].

We used 'quadprog' in Matlab toolbox to solve this problem, then we can get the optimal solution $x^* = (0, 1.0)^T$, and the optimal value $f^* = 4$. We use the algorithm in Section 4 to solve the above problem. Let $x(1) = (-0.4, 1.1)^T$, $\varepsilon = 10^{-5}$, iterate 23 times, the problem is globally converge to an unique optimal solution $x^* = (-0.200003, 0.800004)^T$, and the optimal value of the objective function is $f^* = 3.6405$. Compared with the results in [9], the results in this paper has higher accuracy. Figure.2 show the transient behaviors of the decision variable.

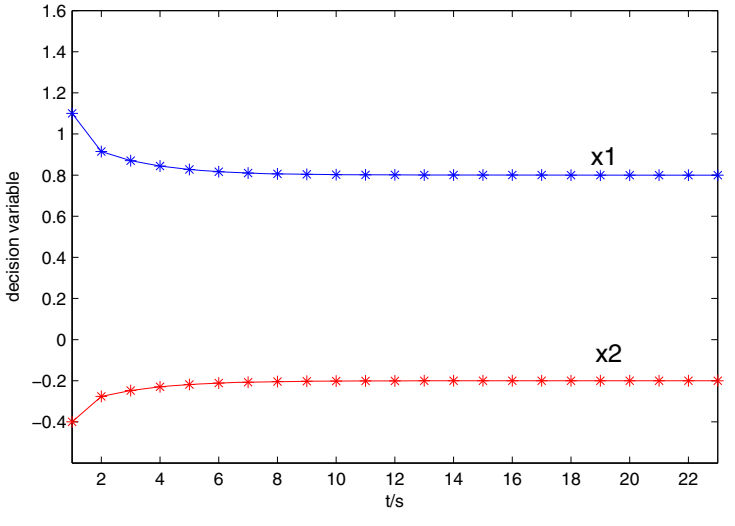


Fig. 2. Transient Behaviors of the decision variable

6 Conclusion

This paper presents a new neural network algorithm for solving quadratic programming problems by using Fibonacci method and neural network. It has also been substantiated that the proposed neural network algorithm is able to generate optimal solution to linear programming with bound constraints. Compared with other neural network models, the objective function of the quadratic programming in this algorithm is universal, which can be convex function or quasi convex function. With the dimension increasing, the advantage of this algorithm is more clearly. It has been shown that the proposed algorithm is easy to implement in computer.

Acknowledgments

This work is supported by the Graduate Scientific Research Creative Foundation of Three Gorges University, China(200949), the Scientific Innovation Team Project of Hubei Provincial Department of Education(T200809), the Natural Science Foundation of Hubei Province, China(2008CDZ046) and the National Natural Science Foundation, China (10726016).

References

1. Horst, R., Pardalos, P.M., Thoai, N.V.: Introduction to Global Optimization. Tsinghua University Publishing House, Beijing (2005)
2. Bertsekas, D.P.: Parallel and Distributed Numerical Methods. Prentice Hall, Englewood Cliffs (1989)
3. Hopfield, J.J., Tank, D.W.: Neural Computation of Decisions in Optimization Problems. Biol. Cybern. 52, 141–152 (1985)
4. Kennedy, M.P., Chua, L.O.: A Neural Networks for Nonlinear Programming. IEEE Trans. on Circuits and Systems 35, 554–562 (1988)
5. Bouzerdoum, A., Pattison, T.R.: Neural Network for Quadratic Optimization with Bound Constraints. IEEE Trans. on Neural Networks 4, 293–304 (1993)
6. Xia, Y.S., Wang, J.: A Recurrent Neural Network for Solving Linear Projection Equation. Neural Networks 13, 337–350 (2000)
7. Xia, Y.S., Wang, J.: A Dual Neural Network for Kinematic Control of Redundant Robot Manipulators. IEEE Trans. Syst.,Man Cybern. Pan B 31, 147–154 (2001)
8. Liang, X.B., Wang, J.: A Recurrent Neural Network for Nonlinear Optimization with a Conntinuously Differentiable Objective Function and Bound Constrains. IEEE Trans. Neural Networks 11, 1251–1262 (2000)
9. Xia, Y.S., Wang, J.: Primal Neural Networks for Solving Convex Quadratic Programs. In: Neural Networks, IJCNN, pp. 582–587 (1999)
10. He, D.X.: Neural Network for Solving Linear Program Based on Bisection Method. Computer Engineering and Applications 20, 102, 74–75 (2006)

A Hybrid Particle Swarm Optimization Algorithm Based on Nonlinear Simplex Method and Tabu Search

Zhanchao Li[1,2], Dongjian Zheng[1,2], and Huijing Hou[1]

[1] College of Water Conservancy and Hydropower Engineering, Hohai University, Nanjing 210098, Jiangsu, China
lizhanchao@gmail.com
[2] State Key Laboratory of Hydrology-Water Resources and Hydraulic Engineering, Hohai University, Nanjing 210098, Jiangsu, China

Abstract. Particle swarm optimization (PSO) algorithm is an intelligent search method based on swarm intelligence. It has been widely used in many fields because of its conciseness and easy implementation. But it is also easy to be plunged into local solution and its later convergence speed is very slow. In order to increase its convergence speed, nonlinear simplex method (NSM) is integrated into it, which not only can increase its later convergence speed but also can effectively avoid dependence on initial conditions of NSM. In order to bring particles jump out of local solution regions, tabu search (TS) algorithm is integrated into it to assign tabu attribute to these regions, which make it with global search ability. Thus the hybrid PSO algorithm is an organic composition of the PSO, NSM and TS algorithms. Finally its basic operation process and optimization characteristics are analyzed through some benchmark functions and its effectiveness is also verified.

Keywords: hybrid algorithm, particle swarm optimization, nonlinear simplex method, tabu search.

1 Introduction

The PSO algorithm [1,2] is still in the preliminary stage since it was proposed in 1995, and there are still many theories of it should to be perfected. It is widely used in function optimization, neural network training, fuzzy system control and so on because of its conciseness, easy implementation, needing to adjust little parameters, not requiring gradient information and other excellent features [3]. But how to increase the convergence speed and how to avoid premature convergence have always been the focus of most researchers and they are also the problems faced by all the other random search algorithms [4]. One of the main directions to improve the PSO algorithm is to establish the hybrid PSO algorithm. Hendtlass [5] combined different evolution with the PSO. Parsopoulos [6] etc initialized the PSO using NSM. Mirnada [7] etc put together the best features of evolution strategies with the PSO. Krink [8] etc introduced a hybrid approach called Life Cycle model that simultaneously applied genetic algorithms, PSO and stochastic hill climbing to create a generally well-performing search heuristics. Shi

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 126–135, 2010.

[9] etc proposed two hybrid evolutionary algorithms based on PSO and GA in parallel and series forms respectively. Noel [10] etc introduced a hybrid PSO making use of gradient information to achieve faster convergence without getting trapped in local minima. Wachowiak [11] etc embedded Powell method in the PSO to improve accuracy. Vietoire [12] etc integrated the PSO technique with the sequential quadratic programming technique to solve the economic dispatch problem.

The hybrid PSO algorithms mentioned above are mainly hybrid algorithms of global optimization algorithms with local optimization algorithms or other global ones. Essence of these hybrid algorithms is using advantages of each algorithm, which allows different algorithms to perform their strengths and avoid their shortcomings, so as to reach equilibrium among them. However, existing hybrid PSO algorithms have hardly both studied the increase of convergence speed and the avoidance of premature convergence at the same time. Moreover, it has also not been studied deeply enough to avoid premature convergence, which is not perfect for a good algorithm. Here it is improved from two aspects: (1) using the NSM which has a strong local search ability to enhance its local search ability in the later stage; (2) using the TS algorithm to deal with the local extremum regions so as to bring the particles within them all out, and at the same time to avoid the low efficiency of the TS algorithm caused by its limitation in dealing with plentiful individuals (particles).

2 Backgrounds of the PSO, NSM and TS Algorithm

2.1 Particle Swarm Optimization (PSO)

In the PSO algorithm, particle i is expressed as $X_i=(x_{i1}, x_{i2},\ldots, x_{iD})$, which represents a point in the D-dimensional solution space. Each particle saves the best position of itself so far $P_i=(p_{i1}, p_{i2},\ldots, p_{iD})$ and its current flying speed $V_i=(v_{i1}, v_{i2},\ldots, v_{iD})$. The best location so far found by the whole particle swarm is $P_g=(p_{g1}, p_{g2},\ldots, p_{gD})$. In each iteration, the particle's flying speed in the D-dimensional space is updated through P_g, P_i and X_i. Then its position is updated through the updated flying speed. The updating formulae of the PSO algorithm are as flows.

$$V_{id}^{New}(t+1) = w\times V_{id}^{old}(t) + c_1\times rand()\times(p_{id}(t) - x_{id}^{old}(t)) + c_2\times rand()\times(p_{gd}(t) - x_{id}^{old}(t)) \tag{1}$$

$$x_{id}^{New}(t+1) = x_{id}^{old}(t) + V_{id}^{New}(t+1) \tag{2}$$

$$v_{\min} \le v_{id} \le v_{\max}, x_{i\min} \le x_i \le x_{i\max} \tag{3}$$

Where c_1 and c_2 are two positive constants called learning factors, which make each particle with the ability of self-summary and learning from the outstanding particles, so as to get close to the best positions of itself and the whole swarm so far; w is the inertia weight factor, which can dynamically adjust the search ability of the particle swarm along with the time; rand()is the random number between 0 and 1, which is used to maintain the diversity of the particle swarm. (1) is the updating formula of the particle

speed, which indicates that each particle updates its speed based on its original speed V_{id}, its best location so far P_{id} and the best location of the whole particle swarm P_{gd}. (2) is the updating formula of the particle position and (3) is the constraints of the particle speed and position.

2.2 Non-linear Simplex Method (NSM)

The NSM was developed by Nelder and Mead [13] based on the basic simplex method. It is a widely used direct local search technology for non-linear unconstrained optimization problems because it can complete the optimization directly according to the function value without its derivative information. Its basic principle is that: first construct a convex polyhedron with N+1 vertices in the N-dimensional Euclidean space E_N, calculate the function value of each vertex and determine the maximum, the second largest value and the minimum; then find a better solution to substitute the maximum through reflection, expansion, contraction and reduction; finally approach a better minimum through many iterations.

2.3 Tabu Search (TS)

The TS algorithm is firstly proposed in 1986 by Glover [14] and is a sub-heuristic search technique. It is a reflection of artificial intelligence and an extension of local search. Its most important thought is that it can mark the searched local minima and can avoid them as much as possible so as to ensure an effective search path. Its basic principle is that: determinate a number of candidates of the current solution in its given neighborhood; if the best candidate is better than the best solution so far, its taboo attribute is neglected and it is used to substitute the current solution and the best solution so far, and the tabu table and the tenure are modified; if the candidate mentioned above does not exist, choose the best one in the non-taboo candidates as the new current solution while ignoring its strengths and weaknesses to the current solution and modify the tabu table at the same time; repeat the above iteration until the end criteria is met.

3 The Hybrid PSO Algorithm

3.1 Hybrid Strategy

The PSO algorithm is suitable for non-linear and multi-extremum optimization problems because of its conciseness, easy implementation, fast calculation, not requirement for the objective function's mathematical form and its gradient information. And also it has better portability, robustness and can compute in parallel. The NSM can complete optimization directly according to the function value without its derivative information. So it is a widely used direct local search technology for non-linear unconstrained optimization problems. Because of its needless of the first derivative, Hessen matrix and complex matrix operations, it is particularly adapted to the optimization of complex functions with incomplete information. However, this method is very sensitive to the

initial conditions, and can not be guaranteed to converge to the global optimal solution. The TS algorithm has a strong "mountain climbing" ability to jump out of the local optimal solution in the search process so as to search in other regions. So the probability to get a better solution or the global optimal solution is greatly increased.

Generally, it is difficult to realize efficient optimization only depending on search algorithms with a single neighborhood structure, and it is an effective means to broaden application scope and improve performance of an algorithm that make the algorithm with a hybrid neighborhood structure [15]. At present, the hybrid structure of an algorithm can be divided into serial, mosaic, parallel and mixed structures. Hybrid algorithm with a serial structure can absorb advantages of different algorithms. For example, it can use results of an algorithm as the starting point of another algorithm to optimize the problem sequentially and its purpose is to improve the optimization efficiency on the premise of a certain good quality. Hybrid algorithm with a mosaic structure performs that an algorithm is an optimization operation of another algorithm or an evaluating device of the search performance. Hybrid of these algorithms is in view of their complementarities so as to overcome the premature convergence and (or) the plunge into local minimum of a single algorithm. And mosaic structure allows the aided algorithm to be executed repeatedly and the information between the sub-algorithms can exchange bidirectional. Therefore, the mosaic structure is applied to the hybrid of the PSO, NSM and TS algorithm.

For global optimization problems, a good algorithm should have a strong global exploration ability to obtain suitable seeds in the earlier stage while a strong local search ability to increase the convergence speed in the later stage, and also can jump out of the local minimum to search in other regions when the obtained solution does not meet accuracy requirements. The PSO algorithm has a faster convergence speed and a better global exploration ability in the earlier stage but a slower convergence speed in the later stage. The main reason is that in the later stage most particles have congregated in the vicinity of the optimal particle and almost stopped moving. But the approximate area of the minimum has been well located which provides good initial conditions for the NSM. Therefore, application of the NSM at this stage can not only strengthen the local search ability of the PSO algorithm, but also avoid the easy plunge of the NSM into the local minimum because of its very dependence on initial conditions. This hybrid algorithm performs well on functions with few local minimums, while has a limited improvement on multimodal functions with lots of local minima and global extremums because even in the simple low-dimensional cases the computational complexity of these functions is quite high and is a typical NP-hard problem. Parospoulos [16,17] etc used "function stretching" technology to make particles jump out of local minima. But transformation of the objective functions may generate pseudo local minima and misleading gradient information. Although the PSO algorithm does not directly use the gradient information, it is convinced to be used in some indirect way [4]. Here the TS algorithm is used to make particles jump out of the local minimum, that is, when the particles are plunged into it, the local minimum region is dealt with by the TS algorithm and thus the particles can keep away from this region to search globally.

3.2 Basic Steps of the Hybrid PSO Algorithm

The main idea of the hybrid PSO algorithm (PSO_NSM_TS) is that: First the particle swarm searches globally in the whole solution space in view of its fast pre-convergence speed and overall exploration ability. Then the first N+ 1 excellent particles are isolated from the swarm in the later stage and the NSM is used to strengthen its local search abilities so as to find the local minimum. When the particle swarm is plunged into a local minimum, the TS algorithm is used to deal with the local minimum region, and particles in this region are re-initialized in the whole solution space. So particles can keep away from this region to search globally in the subsequent search process. The basic steps are as follows:

(1) Initialize the particle swarm: the initial position and velocity of each particle are randomly initialized in the solution space for the given particle swarm with 3N+1 particles;

(2) Calculate fitness of each particle and sort: the fitness of each particle is calculated according to the objective function and all particles are sorted from small to big according to the fitness;

(3) Calculate the average particle distance [18] of the fist N+1 excellent particles $D(t) = \frac{1}{SgL}\sum_{i=1}^{s}\sqrt{\sum_{d=1}^{N}(p_{id} - \overline{p}_d)^2}$, where L is the maximal diagonal length of the search space; S is the swarm size, here taken to be N+1; N is the dimension of the solution space;

(4) Judge the average particle distance: if D(t)≤[D], call the NSM and go to step (5); if D(t)>[D], use the PSO algorithm to update the whole particle swarm and go to step (8);

(5) Call the NSM: the NSM is used among the first N+1 excellent particles to find the local solution;

(6) Judge the local solution: if the local solution searched by the NSM meets accuracy requirement, the whole optimization is completed and exits; if not, call the TS algorithm;

(7) TS algorithm updates the tabu table: add the current local solution region to the tabu table and update the tenure; Because the current speed of the particles in the local minimum region is very slow and in order to bring them to quickly jump out of this region, the particles within the region are re-initialized. Meanwhile, in order to avoid the particle swarm to be plunged into the local minimum region again, the best positions of these particles are also re-initialized and the best position of the swarm is updated correspondingly; In order to organically combine with the PSO algorithm and also not increase the complexity of the hybrid algorithm, poor fitness is given to the particles within the tabu regions during the PSO updating process and the particles are kept away from the tabu regions due to the attraction of the outstanding particles.

(8) Update the particle swarm: update the particle swarm according to the speed and location updating formula, and go to step (2).

The basic flowchart of the hybrid PSO algorithm is shown in Fig. 1.

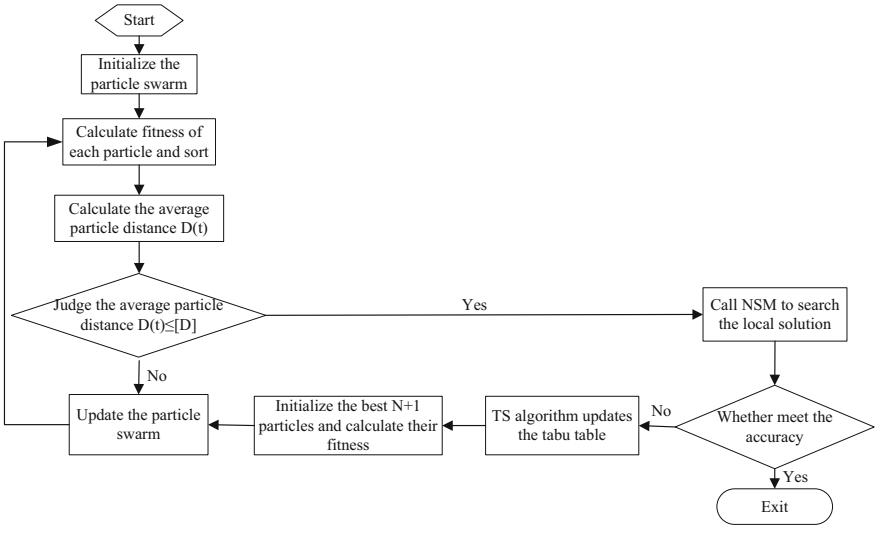


Fig. 1. Flowchart of the hybrid PSO algorithm

4 Optimization Analysis of the Hybrid PSO Algorithm

In order to analyze the basic operation process of the hybrid PSO algorithm, the DeJong function [3] is taken as an example to be optimized. Its expression is $f(x)=\sum_{i=1}^{N} x_i^2$, $-100 \le x_i \le 100$. Its global minimum is min(f)=f(0,…,0)=0. The change process of the best fitness so far along with the iteration of the hybrid PSO algorithm is shown in Fig. 2.

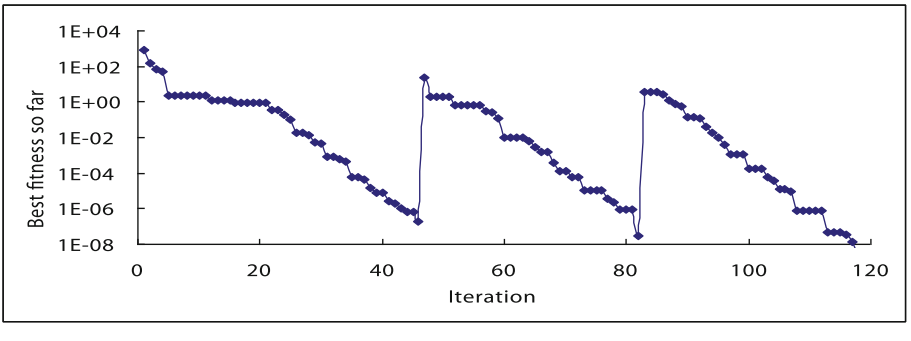


Fig. 2. Change process of the best fitness so far along with the iteration

As can be seen from Fig. 2, in the earlier stage the hybrid PSO algorithm searches in the solution space according to the basic PSO algorithm. And the NSM is called when the first N+1 excellent particles satisfy the condition of D(t)≤[D]. When the solution found by the NSM within the allowed iteration does not meet accuracy

requirement, it is considered to be plunged into the local minimum and the TS algorithm is used to bring the particles jump out of this region.

In order to analyze the characteristics of the hybrid PSO algorithm in dealing with multi-minimum problems, the Rastrigin function is taken as an example. Its expression [3] is $f(x)=\sum_{i=1}^{N}[x_i^2-10\cos(2\pi x_i)+10]$, $-10\le x_i\le 10$. Its global minimum is min(f)=f(0,…,0)=0. Fig. 3 shows the two-dimensional Rastrigin function. This function is tested 50 times using the basic PSO algorithm, the PSO_NSM hybrid algorithm, the PSO_TS hybrid algorithm and the PSO_NSM_TS hybrid algorithm respectively, and the average iterations, the convergence success rate and the average number of calling the TS algorithm of these algorithms are shown in table 1. Fig. 4 shows the change process of the best fitness so far along with the iteration of the basic PSO algorithm and the PSO_NSM hybrid algorithm under convergence and the PSO_NSM_TS hybrid algorithm and the PSO_TS hybrid algorithm under convergence when calling TS.

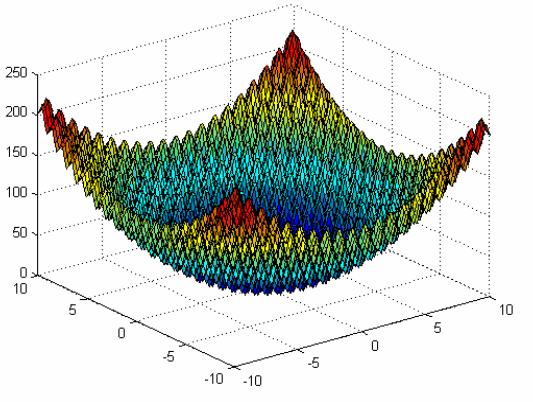


Fig. 3. The two-dimensional Rastrigin function

Table 1. Result of these PSO algorithms

Type	Average iterations[1]	Convergence success rate (%)	Average number of calling TS
PSO_NSM_TS	228.76	100	0.64
PSO_TS	358.28	100	1.18
PSO_NSM	102.66	68	/
PSO	128.26	38	/

[1] The average iterations of the PSO_NSM hybrid algorithm and the basic PSO algorithm are only iterations under convergence.

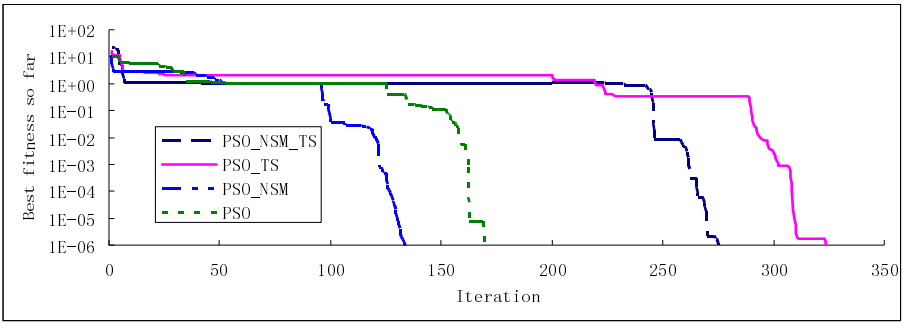


Fig. 4. The best fitness so far along with the iteration of the PSO_NSM_TS, PSO_TS, PSO_NSM and PSO

As can be seen from Table 1 and Fig. 4, for the multi-minimum problems the PSO_NSM hybrid algorithm is the fastest, the basic PSO algorithm second, while the PSO_TS hybrid algorithm is the slowest which indicates that the convergence speed can be increased through the introduction of the NSM. The convergence success rates of the basic PSO algorithm and the PSO_NSM hybrid algorithm are only 38% and 68%, while the convergence success rate can be increased to 100% through the introduction of the TS algorithm. The average number of calling the TS algorithm of the PSO_NSM_TS hybrid algorithm is only 0.64 which is far lower than that of the PSO_TS hybrid algorithm. Furthermore, other benchmark functions are also used to test the proposed hybrid algorithm such as the generalized Rosenbrock function, Griewank function and so on. The result shows that the proposed hybrid algorithm in this paper not only can increase the convergence speed but also can avoid premature convergence, which is the starting point of this paper.

In a word, the key point of the hybrid PSO algorithm (PSO_NSM_TS) is the introduction of the NSM and the TS algorithm. The introduction of the TS algorithm prevents the re-plunge of the particles into the known local minimum regions to a great extent. Strength of the preventability is embodied by the tabu length in the TS algorithm. When the tabu length is set to zero, that is, just initialize the first N+1 excellent particles in the current local minimum region and not memorize the previously searched regions. At the moment, the hybrid algorithm is similar to a completely random algorithm. That is, when the particle swarm is plunged into the local minimum region the first N+1 excellent particles are re-initialized to search in the global solution space. When the tabu length is set to infinite, that is, all the preciously searched local minimum regions are memorized and dealt with by the TS algorithm, so the particles search in other un-searched regions and eventually can be able to find the global minimum solution. Therefore, setting of the tabu length not only can harmonize the global search ability and the search speed, but also can harmonize the possibility to treat the local minimum solution as the global minimum solution.

5 Conclusion

The basic PSO algorithm is improved through accelerating the convergence speed and avoiding premature convergence. The improved hybrid PSO algorithm (PSO_NSM_TS)

is an organic composition of the basic PSO, NSM and TS algorithm. In the hybrid algorithm, the NSM is introduced to strengthen the local search ability of the PSO algorithm in the later stage, and the TS algorithm is introduced to bring particles jump out of the local minimum regions when the particle swarm is plunged into them. So the hybrid PSO algorithm can increase the convergence speed and avoid premature convergence to a great extent. And the benchmark functions also verify its correctness and high efficiency. The pseudo-minima tend to be generated when the tabu regions are dealt with poor fitness. But they only survive within the tabu length. When the local minimum regions which generate them are released, they are automatically eliminated. So using the TS algorithm can solve the pseudo-minima problem to a certain extent. And it can increase the convergence speed and reduce the calculation work when the pseudo-minimum problem is effectively solved. At the same time, there are still some problems should to be deeply studied. For example, the hybrid algorithm only considers the region within the average particle distance [D]. If all regions converging to the local minimum can be found and treated, the re-plunge into the local minimum can be completely avoided, which not only increase the global convergence speed but also can improve the search efficiency.

Acknowledgement

This paper is supported by National Natural Science Foundation of China (Grant Nos. 50879024, 50809025, 50539010), National Science and Technology Supporting Plan (Grant No. 2006BAC14B03), Specialized Research Fund for the Doctoral Program of Higher Education (Grant No. 20070294023) and Graduate Innovation Program of University in Jiangsu Province (Grant No. CX09B_163Z).

References

1. Eberhart, R.C., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
2. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Piscataway, NJ, USA, pp. 1942–1948 (1995)
3. Ji, Z., Liao, H.L., Wu, Q.H.: Particle Swarm Optimization and Its Application. Science Press, Beijing (2009) (in Chinese)
4. Wang, F.: Research on Particle Swarm Algorithm. South West University, Chongqing (2006) (in Chinese)
5. Hendtlass, T.: A Combined Swarm Differential Evolution Algorithm for Optimization Problems. In: Monostori, L., Váncza, J., Ali, M. (eds.) IEA/AIE 2001. LNCS (LNAI), vol. 2070, pp. 11–18. Springer, Heidelberg (2001)
6. Parsopoulos, K.E., Vrahatis, M.N.: Initializing the Particle Swarm Optimizer Using the Nonlinear Simplex Method. In: Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, pp. 216–221. WSEAS Press (2002)
7. Miranda, V., Fonseca, N.: EPSO-Best-of-Two-Worlds Meta-Heuristic Applied to Power System Problems. In: Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA, pp. 1080–1085 (2002)

8. Krink, T., Lovbjerg, M.: The Life Cycle Model: Combining Particle Swarm Optimization, Genetic Algorithms and Hill Climbers. In: Proceedings of Parallel Problem Solving from Nature VII, pp. 621–630 (2002)
9. Shi, X., Lu, Y., Zhou, C., Lee, H., Lin, W., Liang, Y.: Hybrid Evolutionary Algorithms Based on PSO and GA. In: Proceedings of IEEE Congress on Evolutionary Computation (CEC), Canbella, Australia, pp. 2393–2399 (2003)
10. Noel, M.M., Jannett, T.C.: Simulation of a New Hybrid Particles Swarm Optimization Algorithm. In: Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory, pp. 150–153 (2004)
11. Wachowiak, M.P., Smolfkova, R., Zheng, Y., Zurada, J.M., Elmaghraby, A.S.: An Approach to Multimodal Biomedical Image Registration Utilizing Particle Swarm Optimization. IEEE Transactions on Evolutionary Computation 8(3), 289–301 (2004)
12. Victoire, T.A.A., Jeyakumar, A.E.: Hybrid PSO-SQP for Economic Dispatch with Valve-point Effect. Electric Power Systems Research 71(1), 51–59 (2004)
13. Nelder, J.A., Mead, R.: A Simplex Method for Function Minimization. Computer Journal 7, 308–313 (1965)
14. Glover, F.: Future Paths for Integer Programming and Links to Artificial Intelligence. Computers and Operations Research 13, 533–549 (1986)
15. Wang, L.: Intelligence Optimization Algorithm and Its Application. Tsinghua University Press, Beijing (2001) (in Chinese)
16. Parsopoulos, K.E., Magoulas, V.P.G., Vrahatis, M.: Stretching Technique for Obtaining Global Minimizes through Particle Swarm Optimization. In: Proceedings of the workshop on particle swarm optimization, Indianapolis, IN (2001)
17. Parsopoulos, K.E., Vrahatis, M.N.: Modification of the Particle Swarm Optimizer for Locating All the Global Minima. In: Proceeding of the International Conference on Artificial Neural Networks and Genetic Algorithms, Prague, Czech Republic, pp. 324–327 (2001)
18. Riget, J., Vesterstroem, J.S.: A Diversity-guided Particle Swarm Optimizer – the ARPSO. Technical Report, Dept. of Computer Science, University of Aarhus, EVALife No.2002-02 (2002)

Fourier Series Chaotic Neural Networks

Jia-hai Zhang[1], Chen-zhi Sun[1], and Yao-qun Xu[2]

[1] College of Electrical and Automatic Engineering, Sanjiang University, 210012, Nanjing, China
zhangjh6688@sohu.com, sunchenzhi@sju.js.cn
[2] Institute of System Engineering, Harbin University of Commerce, 150028, Harbin, China
xuyq@hrbcu.edu.cn

Abstract. In this paper, Fourier series chaotic neural network model is presented to improve the ability to escape the local minima so that it can effectively solve optimization problems. 10-city traveling salesman problem was given and the effects of the non-monotonous degree in the model on solving 10-city traveling salesman problem were discussed, the figures of the reversed bifurcation and the maximal Lyapunov exponents of single neural unit were given. The new model is applied to solve several function optimizations. Seen from the simulation results, the new model is powerful than the common chaotic neural network.

1 Introduction

The Hopfield network, proposed by Hopfield and Tank [1], has been extensively applied to many fields in the past years. Several chaotic neural networks with non-monotonous activation functions have been proved to be more powerful than Chen's chaotic neural network in solving optimization problems, especially in searching global minima of continuous function and traveling salesman problems [2, 7~8]. The reference [3] has pointed out that the single neural unit can easily behave chaotic motion if its activation function is non-monotonous. And the reference [4] has presented that the effective activation function may adopt kinds of different forms, and should embody non-monotonous nature. The chaotic mechanism of this new model is introduced by the self-feedback connection weight. The activation function of the new chaotic neural network model is composed of Sigmoid and trigonometric function, therefore the activation function is non-monotonous. And because trigonometric function is a kind of basic function, the model can solve optimization problems more effectively. Finally, the new model is applied to solve both function optimizations and combinational optimizations and the effects of the non-monotonous degree in the model on solving 10-city TSP are discussed. Seen from the simulation results, the new model is powerful than the common chaotic neural network.

2 Fourier Series Chaotic Neural Network (FSCNN)

Fourier series chaotic neural network is described as follows:

$$x_i(t) = f(y_i(t)) \tag{1}$$

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 136–143, 2010.

$$y_i(\mathrm{t}+1) = ky_i(\mathrm{t}) + \alpha\left[\sum_{\substack{j=1 \\ j\neq i}}^{n} w_{ij}x_j(\mathrm{t}) + I_i\right] - z_i(\mathrm{t})(x_i(\mathrm{t}) - I_0) \tag{2}$$

$$z_i(\mathrm{t}+1) = (1-\beta)z_i(\mathrm{t}) \tag{3}$$

$$f(u) = \mathrm{S}_1(u) + \mathrm{S}_2(u) \tag{4}$$

$$\mathrm{S}_1(u) = \frac{1}{1+\exp(-u/\varepsilon_0)} \tag{5}$$

$$\mathrm{S}_2(u) = \omega_1\cos(\varepsilon_1 u) + \omega_2\sin(\varepsilon_2 u) \tag{6}$$

Where i is the index of neurons and n is the number of neurons, $x_i(t)$ the output of neuron i, $y_i(t)$ the internal state for neuron i, W_{ij} the connection weight from neuron j to neuron i, I_i the input bias of neuron i, α the positive scaling parameter for inputs, k the damping factor of the nerve membrane ($0 \le k \le 1$), $z_i(t)$ the self-feedback connection weight, $\varepsilon_0, \varepsilon_1, \varepsilon_2$ the steepness parameters of the activation function, β the simulated annealing parameter of the self-feedback connection weight $z_i(t)$, I_0 a positive parameter and ω_1, ω_2 the parameters of the trigonometric function.

In this model, the variable $z_i(t)$ corresponds to the temperature in the usual stochastic annealing process and the equation (3) is an exponential cooling schedule for the annealing. The chaotic mechanism is introduced by the self-feedback connection weight as the value of $z_i(t)$ becomes small step by step. The chaotic behavior plays a global search role in the beginning. When the value of $z_i(t)$ decreases to a certain value, the network functions in a fashion similar to the Hopfield network which functions in gradient descent dynamic behavior. Finally, the neurons arrive at a stable equilibrium state. The reference [5] shows that both the parameter β governed the bifurcation speed of the transient chaos and the parameter α could affect the neuron dynamics; in other words, the influence of the energy function was too strong to generate transient chaos when α was too large, and the energy function could not be sufficiently reflected in the neuron dynamics when α was too small. So in order for the network to have rich dynamics initially, the simulated annealing parameter β must be set to a small value, and α must be set to a suitable value, too.

In this model, the parameter ω_1, ω_2 presents the non-monotonous degree of the activation function. Seen from the equations (4) and (5), it is concluded that the equation (4) is similar to the function of Sigmoid alone in form in the circumstance of the value of ω_1, ω_2 being between 0 and 1 without consideration of the monotonous nature. So the parameter ω_1, ω_2 presents a local non-monotonous phenomenon of the activation function. In other words, if the parameter ω_1, ω_2 borders on 1, the non-monotonous

phenomenon of the activation function is very apparent; otherwise, if the parameter ω_1, ω_2 borders on 0, the non-monotonous phenomenon of the activation function is very weak.

3 Research on Single Neural Unit

In this section, we make an analysis of the neural unit of the Fourier series chaotic neural networks.

The single neural unit can be described as (7) ~ (9) together with (4) ~ (6):

$$x(t) = f(y(t)) \tag{7}$$

$$y(t+1) = ky(t) - z(t)(x(t) - I_0) \tag{8}$$

$$z(t+1) = (1-\beta)z(t) \tag{9}$$

In order to make the neuron behave transient chaotic behavior, the parameters are set as follows:

$$\varepsilon_0 = 0.02, \varepsilon_1 = 2, \varepsilon_2 = 2, \omega_1 = 1/3, \omega_2 = 1/3, y(1) = 0.283, z(1) = 0.4, k = 1, I_0 = 0.65$$

The state bifurcation figures and the time evolution figures of the maximal Lyapunov exponent are respectively shown as Fig.1~Fig.4 when $\beta = 0.004$ and $\beta = 0.002$.

Seen from the above state bifurcation figures, the neuron behaves a transient chaotic dynamic behavior. The single neural unit first behaves the global chaotic search, and with the decrease of the value of $z(0,0)$, the reversed bifurcation gradually converges to a stable equilibrium state. After the chaotic dynamic behavior disappears, the dynamic behavior of the single neural unit is controlled by the gradient descent dynamics. When the behavior of the single neural unit is similar to that of Hopfield, the network tends to converge to a stable equilibrium point. The simulated annealing parameter β affects the length of the reversed bifurcation, that is, the smaller value of β prolongs the reversed bifurcation.

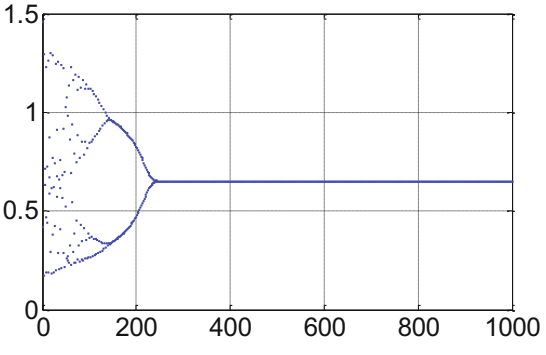


Fig. 1. State bifurcation figure of the neuron when $\beta = 0.004$

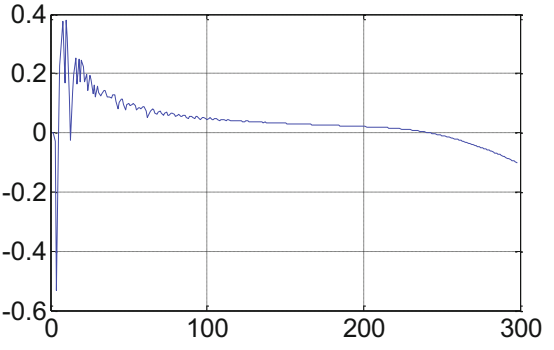


Fig. 2. Time evolution figure of the maximal Lyapunov exponent of the neuron when β=0.004

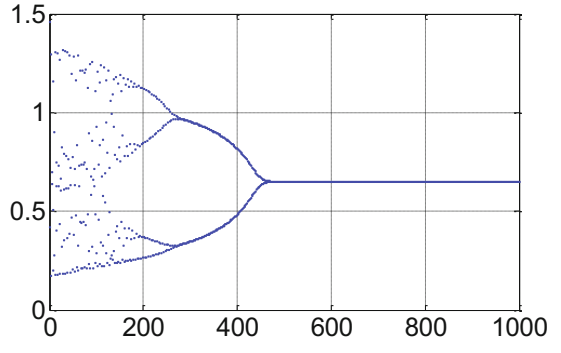


Fig. 3. State bifurcation figure of the neuron when β=0.002

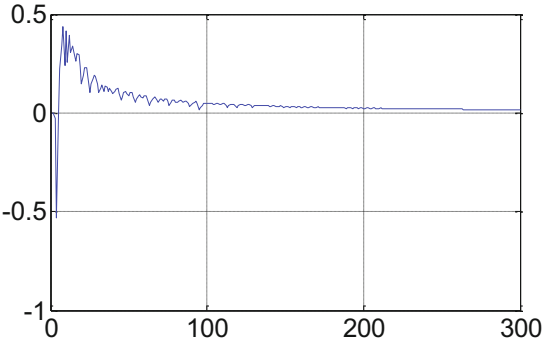


Fig. 4. Time evolution figure of the maximal Lyapunov exponent of the neuron when β=0.002

4 Application to Continuous Function Optimization Problems

In this section, we apply the Fourier series chaotic neural network to search global minima of the following function.

The function is described as follows [6]:

$$f_2(x_1, x_2) = (x_1 - 0.7)^2[(x_2 + 0.6)^2 + 0.1] + (x_2 - 0.5)^2[(x_1 + 0.4)^2 + 0.15] \quad (10)$$

The minimum value of (10) is 0 and its responding point is (0.7, 0.5).

The parameters are set as follows:

ε_0 =2.5, ε_1 =20, ε_2 =10, k =1, ω_1 =0.1, ω_2 =0.05, $y_1(1) = y_2(1)$ =0.283, β =0.05, α =0.4, $z_1(1) = z_2(1)$ =0.3, I_0 =0.65.

The time evolution figure of the energy function of FSCNN in solving the function is shown as Fig.5.

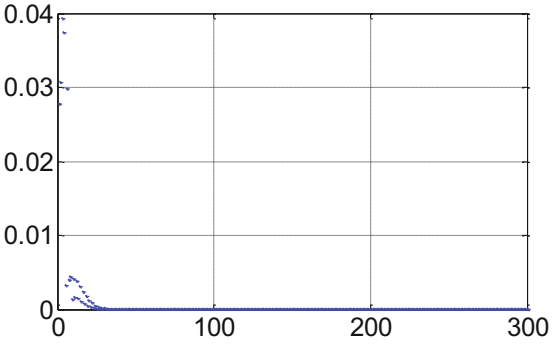


Fig. 5. Time evolution figure of energy function

The global minimum and its responding point of the simulation are respectively 1.3236e-016 and (0.7, 0.5).

This section indicates that FSCNN has a good performance to solve function optimization problems. In order to testify the performance of FSCNN, the new model is applied to solve 10-city traveling salesman problems.

5 Application to 10-City TSP

A solution of TSP with N cities is represented by N×N-permutation matrix, where each entry corresponds to output of a neuron in a network with N×N lattice structure. Assume V_{xi} to be the neuron output which represents city x in visiting order i. A computational energy function which is to minimize the total tour length while simultaneously satisfying all constrains takes the follow form:

$$E = \frac{A}{2}\sum_{x=1}^{n}(\sum_{i=1}^{n} V_{xi} - 1)^2 + \frac{B}{2}\sum_{i=1}^{n}(\sum_{x=1}^{n} V_{xi} - 1)^2 + \frac{D}{2}\sum_{x=1}^{n}\sum_{y=1}^{n}\sum_{i=1}^{n} d_{xy} V_{xi} V_{y,i+1} \quad (11)$$

A and B (A=B) are the coupling parameters corresponding to the constrains and the cost function of the tour length, respectively. d_{xy} is the distance between city x and city y.

This paper adopts the following 10-city unitary coordinates:
(0.4, 0.4439), (0.2439, 0.1463), (0.1707, 0.2293), (0.2293, 0.716), (0.5171, 0.9414), (0.8732, 0.6536), (0.6878, 0.5219), (0.8488, 0.3609),(0.6683, 0.2536), (0.6195, 0.2634). The shortest distance of the 10-city is 2.6776.

The reference [5] has presented that the effective activation function may adopt kinds of different forms, and should behave non-monotonous behavior. In this paper, ω_1, ω_2 that represents the non-monotonous degree is analyzed in order to simply ascertain the effect of the non-monotonous degree to FSCNN in solving 10-city TSP. Therefore, the models with different values of ω_1, ω_2 in solving 10-city TSP are analyzed as follows:

The parameters of the network are set as follows:

ε_1=20, ε_2 =10, ε_0 =1/30, k =1, α =0.6, $z(1)$ =0.1, I_0 =0.2,A=1.4, D=1.5,△t=0.04.

2000 different initial conditions of y_{ij} are generated randomly in the region [0, 1] for different β. The results are summarized in Table1, the column 'NL', 'NG', 'LR' and 'GR' respectively represents the number of legal route, the number of global optimal route, the rate of legal route, the rate of global optimal route.

The lager value of the simulated annealing parameter β is regarded stronger if the network can all converge to the global minimum in 2000 different random initial conditions.

Seen from table 1, the follow observations can be drawn according to numerical simulation test:

First, the model with smaller ω_1, ω_2 s such as ω_1 =0.01, ω_2 =0.01; ω_1 =0.02, ω_2 =0.01; ω_1 =0.025, ω_2 =0.01; ω_1 =0.01, ω_2 =0.025; ω_1 =0.02, ω_2 =0.02 in solving 10-city TSP can all converge to the global minimum. But, it is not true that the smaller the parameter ω_1, ω_2 is, the more powerful the ability to solve 10-city. Because, for example, the parameter ω_1 =0.02, ω_2 =0.02 can all converge to the global minimum as β =0.001 while the parameter ω_1 =0.01, ω_2 =0.01 can almost converge to the global minimum as β =0.0008.

Second, with the decrease of the value of ω_1 and ω_2, the value of 'NG' becomes large gradually from 1893(ω_1 =0.02, ω_2 =0.02) to 1973(ω_1 =0.01, ω_2 =0.005) as β =0.01 .In other word, with the decrease of the value of ω_1 and ω_2, the ability to get global optimal route becomes strong.

Table 1. Results of 2000 different initial conditions for each value β on 10-city TSP

ω_1, ω_2	β	NL	NG	LR	GR
$\omega_1=0.001, \omega_2=0.001$	0.01	2000	1953	100%	97.65%
	0.008	2000	1983	100%	99.15%
	0.001	2000	1994	100%	99.7%
	0.0008	2000	1995	100%	99.75%
$\omega_1=0.005, \omega_2=0.01$	0.01	2000	1972	100%	98.6%
	0.008	2000	1994	100%	99.7%
	0.001	2000	1994	100%	99.7%
	0.0008	2000	1998	100%	99.9%
$\omega_1=0.01, \omega_2=0.01$	0.01	2000	1984	100%	99.2%
	0.008	2000	1998	100%	99.9%
	0.001	2000	1998	100%	99.9%
	0.0008	2000	2000	100%	100%
$\omega_1=0.02, \omega_2=0.01$	0.01	2000	1955	100%	97.75%
	0.008	2000	1945	100%	97.25%
	0.001	2000	1996	100%	99.8%
	0.0008	2000	2000	100%	100%
$\omega_1=0.025, \omega_2=0.01$	0.01	2000	1931	100%	96.65%
	0.008	2000	1931	100%	96.55%
	0.001	2000	2000	100%	100%
	0.0008	2000	2000	100%	100%
$\omega_1=0.01, \omega_2=0.005$	0.01	2000	1973	100%	98.65%
	0.008	2000	1994	100%	99.7%
	0.001	2000	1995	100%	99.75%
	0.0008	2000	1995	100%	99.75%
$\omega_1=0.01, \omega_2=0.02$	0.01	2000	1940	100%	97%
	0.008	2000	1928	100%	96.4%
	0.001	2000	1998	100%	99.9%
	0.0008	2000	1997	100%	99.85%
$\omega_1=0.01, \omega_2=0.025$	0.01	2000	1908	100%	95.4%
	0.008	2000	1923	100%	96.15%
	0.001	2000	2000	100%	100%
	0.0008	2000	2000	100%	100%
$\omega_1=0.02, \omega_2=0.02$	0.01	2000	1893	100%	94.65%
	0.008	2000	1927	100%	96.35%
	0.001	2000	2000	100%	100%
	0.0008	2000	2000	100%	100%

Third, when the parameter $\omega_1=0.01, \omega_2=0.025$ and $\omega_1=0.025, \omega_2=0.01$, the ability to all converge to the global minimum is more powerful, that is, the non-monotonous degree of the activation function has a effective on the solution of 10-city TSP.

However, as is analyzed in second, the ability in reaching 'NG' when the parameter $\omega_1=0.01, \omega_2=0.025$ and $\omega_1=0.025, \omega_2=0.01$ is weaker than that of $\omega_1=0.01, \omega_2=0.01$ when $\beta=0.01$ and $\beta=0.008$. So, which model is needed is connected with the concrete request. However, in order to get the tradeoff effect, the value of $\omega_1=0.01, \omega_2=0.01$ may be chose.

6 Conclusion

The presented chaotic neural network called FSCNN is proved to be effective in solving optimization problems, and in the section of application to 10-city TSP, the model with different ω_1 , ω_2 is analyzed and made a comparison. As a result, the simple rule of the model is disclosed. However, there are a lot of questions in the model needed to research. For example, whether does the model in solving 30 or 60 cities' TSP accord with the same rule or so? And different networks have different parameters in solving TSP effectively, is there any way to weigh the different networks in the same parameters?

References

1. Hopfield, J.: Neural Networks and Physical Systems with Emergent Collective Computational Abilities. Proc. Natl. Acad. Sci. 79, 2554–2558 (1982)
2. Xu, Y.-q., Sun, M., Duan, G.-r.: Wavelet Chaotic Neural Networks and Their Application to Optimization Problems. LNCS, vol. 3791, pp. 379–384. Springer, Heidelberg (2006)
3. Potapove, A., Kali, M.: Robust chaos in neural networks. Physics Letters A 277(6), 310–322 (2000)
4. Shuai, J.W., Chen, Z.X., Liu, R.T., et al.: Self-evolution Neural Model. Physics Letters A 221(5), 311–316 (1996)
5. Chen, L., Aihara, K.: Chaotic Simulated Annealing by a Neural Network Model with Transient Chaos. Neural Networks 8(6), 915–930 (1995)
6. Ling, W.: Intelligence optimization algorithm and its application. Press of TUP (2001)
7. Xu, Y.-q., Sun, M.: Gauss-Morlet-Sigmoid Chaotic Neural Networks. In: Huang, D.-S., Li, K., Irwin, G.W. (eds.) ICIC 2006. LNCS, vol. 4113, pp. 115–125. Springer, Heidelberg (2006)
8. Xu, Y.-q., Sun, M., Shen, J.-h.: Gauss Chaotic Neural Networks. In: Yang, Q., Webb, G. (eds.) PRICAI 2006. LNCS (LNAI), vol. 4099, pp. 319–328. Springer, Heidelberg (2006)

Multi-objective Optimization of Grades Based on Soft Computing

Yong He

School of Management, Guangdong University of Technology, Guangzhou
510520 China
jiabeisheng@sina.com

Abstract. Economic benefit and resource utilization rate should be considered when optimizing the cut-off grade and grade of crude ore in production management of metal mine. This paper introduces soft computing to the field of mine, to determine the combination of grades in the condition of multiple objectives, and its basic idea is that, ANN is used to model the nonlinear function from the relative variables to metal recovery of milling and total cost, fuzzy comprehensive evaluation integrates multiple objection, GA searches for the optimal grades combination, these three techniques not only function independently but also effectively integrate together, collectively display the function of modeling, reasoning and optimization. Take Daye Iron mine as an example, and it indicated the validity of proposed method, from January to November in 2007, the optimal cut-off grade and grade of crude ore are 16.53% and 43.14%, respectively, and contrasted to the present scheme, lower cut-off grade and higher grade of crude ore can produce more amount of the concentrate ore, get more profit, and enhance higher utilization rate of resource.

Keywords: Multi-objective optimization, Soft computing technique, Cut-off grade, Grade of crude ore.

1 Introduction

In production and management of mining enterprises, we are not only in the pursuit of economic interests, but also take into account of the resource recovery and other relative objectives. Cut-off grade and grade of crude ore are two key indicators in production and management of mine, the optimization is a multi-objective optimization problem. Tao Yong [1], Zhao Dexiao [2], Jiang Annan [3], etc. optimized the grades considering economic benefit with single-objective optimization. However, with the changes from extensive to intensive of mining enterprises and the shortage of resources, we should not only consider economic benefit, but also emphasize on the utilization of resources in the production practice. Li Keqing [4,5], Yuan Huaiyu [6], YU Weijian [7], Li Keqing [8] tried to work on multi-objectives optimization, but the decision-making variables were not the combination of cut-off grade and grade of crude ore. Cut-off grade and grade of crude ore are two concepts related with income, cost, geological grade, loss rate and dilution rate. The mapping function between them

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 144–151, 2010.

are highly-complex and highly-nonlinear, and it is difficult to directly or indirectly find out the mathematics expression.

Soft Computing (SC) [9] is a novel method to create complex system, in order to deal with complex problems under the uncertain and imprecise environment, a variety of different sources of knowledge, technology and methods need to composite and collaborate for computation. Soft computing mainly contains three parts: artificial neural network (ANN) is responsible for pattern cognition and adaptive modeling by changing environment; fuzzy logic (FL) is for reasoning and decision-making of human knowledge; genetic algorithm (GA) leads the system to reach optimization. These three techniques not only function independently but also effectively integrate together, collectively display the function of modeling, reasoning and optimization. In recent years, soft computing has been successfully used in optimization, evaluation and forecast in the fields of engineering technology, economic management, and so on. In this paper, soft computing technique has been used to optimize cut-off grade and grade of crude ore with multiple objectives, to enhance the overall benefit of metal mine system.

2 Set the Objectives

The optimization of cut-off grade and grade of crude ore is based on the economic benefit and resource utilization. The objectives are net present value (NPV), resource utilization rate and total amount of concentrate, respectively. According to mining and milling production process, we set three objectives as follows:

$$MaxNPV = \sum_{t=1}^{n} \frac{R_t - C(a_t, q_t, a_j, a_r)}{(1+i)^t}$$

$$\text{where, } R_t = \frac{a_t q_t (1-\phi(a_j))\varepsilon(a_t, q_t, a_j, a_r)}{\beta} J, 0 \le a_j \le a_r \le 1 \tag{1}$$

$$MaxUTI = \frac{\sum_{t=1}^{n} a_t q_t (1-\phi(a_j))\varepsilon(a_t, q_t, a_j, a_r)}{\sum_{t=1}^{n} a_t q_t} \tag{2}$$

$$\text{where, } 0 \le a_j \le a_r \le 1$$

$$MaxCON = \sum_{t=1}^{n} \frac{a_t q_t (1-\phi(a_j))\varepsilon(a_t, q_t, a_j, a_r)}{\beta} \tag{3}$$

$$\text{where, } 0 \le a_j \le a_r \le 1$$

Where, NPV denotes net present value, R_t denotes revenue of the ore body in the tth month, a_t denotes geological grade in the tth month, a_j denotes cut-off grade,

a_r denotes grade of crude ore, q_t denotes geological reserves in the t th month, $C(a_t, q_t, a_j, a_r)$ denotes the function of cost including mining and milling, $\phi(a_j)$ denotes the function of loss rate related to cut-off grade, $\varepsilon(a_t, q_t, a_j, a_r)$ denotes the function of metal utilization rate of milling, β denotes the concentrate grade, J denotes the current price of the concentrate ore, i denotes the rate of discount.

We can see that, $a_t q_t$ denotes the amount of iron metal of ore body; $1-\phi(a_j)$ denotes mining recovery; $a_t q_t(1-\phi(a_j))$ denotes the amount of iron metal of ore milling outside; $a_t q_t(1-\phi(a_j))\varepsilon(a_t, q_t, a_j, a_r)$ denotes the amount of iron metal in the concentrate powder; $\frac{a_t q_t(1-\phi(a_j))\varepsilon(a_t, q_t, a_j, a_r)}{\beta}$ is the amount of concentrate. That is easy to see, R_t in Eq. (1) denotes revenue, which is the value of the amount of concentrate multiplied by the price of concentrate. The numerator of UTI in Eq. (2) is the total of iron metal in the concentrate in t months, the denominator of Eq. (2) is the amount of iron metal in natural ore body, so UTI denotes the total metal recovery of mining and milling, i.e. the utilization rate of resource. Eq. (3) is the sum of the amount of concentrate.

Obviously, there exist three unknown functions in the three objectives as follows:

Function 1: Mapping relationship between cut-off grade a_j and loss rate ϕ.

Function 2: Mapping relationship between the metal utilization rate ε and cut-off grade a_j, crude ore grade a_r, geological grade a_t, geological reserves q_t.

Function 3: Mapping relationship between the cost of mining & milling C and cut-off grade a_j, crude ore grade a_r, geological grade a_t, geological reserves q_t.

This multi-objective optimization problem is that, given the geological reserves and geological grade, we search for a combination of cut-off grade and grade of crude ore, i.e. the pareto optimal solution, to make trade-offs of three objectives.

3 Establish Non-linear Functions Using Neural Networks

Artificial neural network (ANN) is a parallel non-linear adaptive system composed of a large number of simple processing units, and it originates the learning mechanism of human brain. Without a priori knowledge, ANN is widely used in construct the complex nonlinear system. BP network and RBF networks are two common neural networks [10]. BP network is a multi-layer forward neural network proposed by David Rumelhart and James McClelland in Stanford University [11], and it is named by back-propagation algorithm [12]. The learning process contains the forward propagation of the signal and back-propagation of errors. Radical Basis Function (RBF) neural network [13] was proposed by J. Moody and C. Darken in the late of the 1980s, and it consists of the input layer, hidden layer and output layer. RBF network uses

radial basis function as the basis function of hidden layer neurons, to constitute a hidden layer space, and realizes the input-output mapping. To optimize cut-off grade and grade of crude ore, the key points are to establish the functions from cut-off grade, grade of crude ore to the amount of concentrate, NPV, and the resource utilization rate, and it need to establish the following model: (1) the function of loss rate and cut-off grade; (2) the computing model of metal recovery of milling; (3) the computing model of total cost.

4 Fuzzy Comprehensive Evaluation Integrates Multiple Objectives

Optimization of ore grades is to enhance economic benefit and resource utilization, and the objectives contain profit, the amount of concentrate and resource utilization rate. On one hand, as a result of confliction between the objectives, the impact of decision-making variables on each objective is not synchronized, the impact of the decision-making variables on one objective is positive, may also negatively affect another decision-making objective [14]. For example, with the increase of cut-off grade, grade of crude ore may increase, it leads to enhance economic benefit, but ore recovery may reduce, it causes to reduce the total resource utilization rate. On the other hand, because of incommensurability of different objectives, they have different units, such as the unit of total profit is ten thousand yuan, but the unit of the total concentrate is 10,000 tons. Therefore, all objective value should be normalized to make a reasonable evaluation.

From above, we can see that the production benefit is the amount of concentrate, the economic benefit is NPV, and resource utilization benefit is total metal recovery. For a combination of grades, calculate the above three objectives, then get the fuzzy comprehensive evaluation value, which is used to search the optimal cut-off grade and grade of crude ore. First, define the set of scheme $X = \{scheme\,1, Scheme\ 2, \cdots schemeN\}$, here, each scheme is a combination of cut-off grade and grade of crude ore, and the set of objectives $U = \{$ amount of concentrate, NPV, total metal recovery $\}$, R is fuzzy relation matrix, evaluation space $W = \{X, U, R\}$. The greater the value of three objectives, the greater the contribution to the comprehensive evaluation value, the definition of membership function is as follows:

$$\mu_i(x) = \begin{cases} 0 & x \le x_{\min} \\ (x - x_{\min})/(x_{\max} - x_{\min}) & x_{\min} \le x \le x_{\max} \\ 1 & x \ge x_{\max} \end{cases} \tag{4}$$

We first preset the minimum and maximum of amount of concentrate, NPV and resource utilization rate, respectively, then calculate the membership value of each objective for every scheme by Eq.(4), given the weight value of each objective $\lambda_i\,(i = 1,2,3)$, finally compute the evaluation value by Eq.(5).

$$Z_i = \sum_{i=1}^{3} \lambda_i \mu_i(x) \tag{5}$$

5 Grade Combination Optimization by GA

Genetic algorithm originates biological genetics, and simulates the mechanism of natural selection, evolution and mutation. It aims to improve each individual's fitness, and ultimately find out the optimal individual, which is the best solution. The advantage of GA is fast convergence, global optimization, and it is used widely in optimization of complex nonlinear systems [15]. For a scheme (combination of cut-off grade and grade of crude ore), we calculate the weighted value of the objectives as the individual's fitness, to evaluate the cut-off grade and grade of crude ore. The steps of the algorithm are as follows:

Step1: Initialize the population, and then set the size of population and encoding type.
Step2: Give operators of selection, crossover, mutation and relative parameters.
Step3: For each chromosome, implement Step3.1, Step3.2, Step3.3, Step3.4 and Step3.5.

Step3.1 Decode individuals to the real values of cut-off grade and grade of crude ore.

Step3.2 Calculate NPV of the corresponding grade combination by Eq. (1).

Step3.3 Calculate the corresponding resource utilization rate by Eq. (2).

Step3.4 Calculate the amount of concentrate by Eq. (3).

Step3.5 Compute the fuzzy comprehensive evaluation value of this individual, as its fitness value according to Eq. (5).

Step4: Implement the GA operations of selection, crossover and mutation, and generate new population.
Step5: Check whether reaches the maximum of iterative step or population converges, if not, then switch to Step3.
Step6: Output the best grade combination, and calculate the amount of concentrate, NPV and resource utilization rate according to the best grade combination.

6 Case Study

According to geology, production and cost report forms from Jan.2005 to Nov.2007, we acquire the relative data which are omitted in this paper.

The correlation coefficient of loss rate and cut-off grade is 0.97419, so there is a significant linear relationship, and the function of loss rate and cut-off grade can be expressed as $\phi = 1.7a_j - 12$. The sample simulation diagram is shown in Fig.1. We establish the BP network and RBF networks mapping relationship from cut-off grade, grade of crude ore, geological reserves, geological grade to the metal recovery rate of milling and the total cost. The sample simulation diagram is shown in Fig.2 and Fig.3. It shows that the constructed BP and RBF networks have prefect simulation performance.

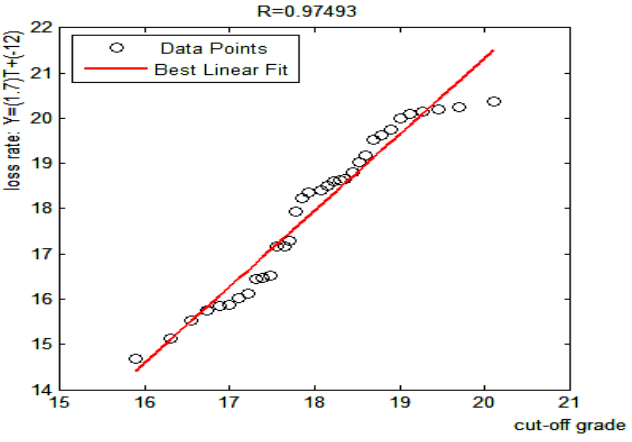


Fig. 1. Regression fit diagram

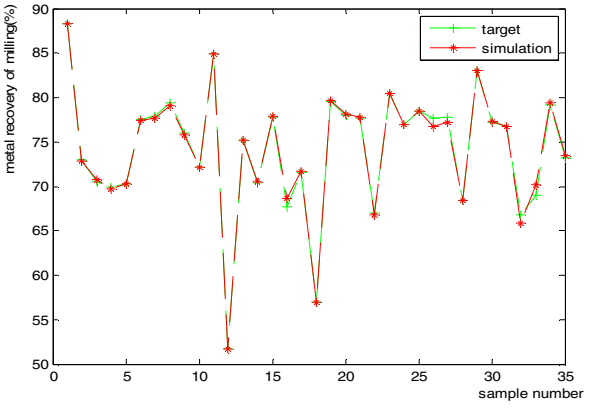


Fig. 2. Metal recovery simulation

Given the geological data and relative grade information, we calculate the loss rate, metal recovery of milling and the total cost, and then calculate NPV, resource utilization rate and the total amount of concentrate, respectively. We normalize these objective values to [0,1], and calculate the weighted value by the weight factors as the fitness value of a certain combination of cut-off grade and grade of crude ore.

Set the price of concentrate ore being 8.08 yuan per ton, the grade of concentrate being 64%, discount rate being 0.005. We determine the weight of decision-making objectives by Delphi, with NPV of 0.4, resource utilization rate of 0.4, the amount of concentrate of 0.2. The population structure of genetic algorithm is that, using binary encoding, with the number of chromosome being 80, the range of cut-off grade being 15-21%, the range of crude ore grade being 40-47%, crossover probability being 0.7, and mutation probability being 0.008.

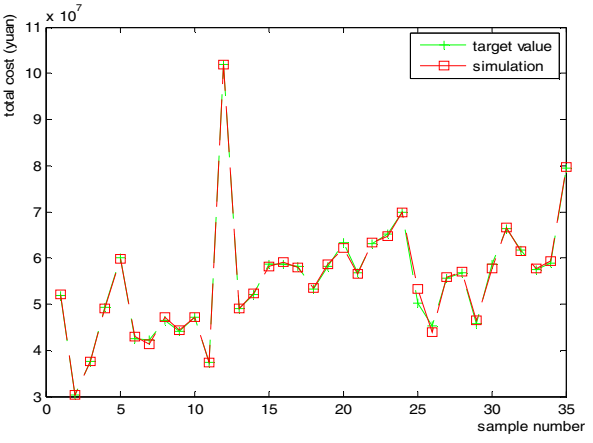


Fig. 3. Total cost simulation

According to geological reserves and grade form January 2007 to November 2007 in Daye iron mine, we optimize the cut-off grade and grade of crude. The results and comparison with the current scheme are as shown in Table1, scheme 1 is the optimized scheme, scheme 2 is the current scheme. Form Table1, we can see that, lower cut-off grade and higher grade of crude ore can produce more amount of the concentrate ore, get more profit, and enhance higher utilization rate of resource. If the cut-off grade is 16.53%, grade of crude ore is 43.14%, from January to November of 2007, the total profit is 113.282 million yuan, the amount of concentrate is 843.3 thousand tons, and the total metal utilization rate is 67.12%. Compared with the current scheme (cut-off grade is 18%, grade of crude ore is 41-42%), the optimized scheme can increase the total profit by 5415 thousand yuan, increase the total concentrate by 110,830 tons, and raise the utilization rate of resource by 5.52 percent.

Table 1. Optimization result and comparison

Scheme	Cut-off grade (%)	Grade of crude ore (%)	Amount of concentrate (10^4ton)	NPV (10^7yuan)	Resource utilization rate (%)
1*	16.53	43.14	84.33	11.3282	67.12
2	18	41-42	73.247	10.7867	61.6

7 Conclusions

In this paper, we take into account of economic benefit and resource utilization benefit, establish multi-objective model of optimizing cut-off grade and grade of crude ore, and use soft computing method to solve it. The result of this research has been applied to Daye iron mine, and significantly improves the benefit. The proposed method of optimizing grades provides a brand-new idea to the metal mine system,

given the input of geological grade, recoverable reserves, metallurgical grade standard, concentrate price, the system automatically outputs the optimal cut-off grade and grade of crude ore, to guide the production of mining and milling. It changes the extensive statement of relying on experience of first-line worker and experiment data, and has broad application value.

Acknowledgments

This research was supported by Doctoral set-up foundation of Guangdong University of Technology, Guangdong Planning Project of Philosophy and Social science grant NO: 09O-06, National Natural Science Foundation grant NO: 70573101 of the People's Republic of China and Research fund grant NO: 070429 of Wuhan Steel and Iron (Group) Corp.

References

1. Tao, Y., Li, W.C.: The optimization of cut-off grade in Wei Steel. Metal Mine 7, 23 (1995)
2. Zhao, D.X., Liu, X.B.: The mathematics model of optimizing dynamically grade of crude ore in Jinshandian Iron Mine. Metal Mine 4, 8–10 (2004)
3. Jiang, A.N., Zhao, D.X.: The application of evolutionary neural network in the optimization of grade of crude ore. Mine industry research and development 24, 44–46 (2004)
4. Li, K.Q., Yuan, H.Y.: Comprehensive evaluation of reasonable grade of crude ore in Waitoushan Iron Mine. Metal Mine 12, 22–25 (1998)
5. Li, K.Q., Yuan, H.Y.: Dynamic optimization method and application of reasonable grade of crude ore. Coloured metal mine exploration 9, 24–27 (2000)
6. Yu, H.Y., Liu, B.S.: Dynamic optimization of reasonable grade of crude ore. Journal of Beijing university of science and technology 24, 239–242 (2002)
7. Yu, W.J., Liu, A.H.: The research of dynamic optimization grade of crude ore in metal mine. China tungsten industry 19, 27–30 (2004)
8. Li, K.Q., Niu, J.K.: The optimization of magnetite grade indexes in Baiyunerbo. Journal of Beijing university of science and technology 29, 334–337 (2007)
9. Guo, S.T., Chen, G.: Soft computing method in information science. Northeast University Press, Shenyang (2001)
10. Jiao, L.C.: Theory of neural network system. Xidian Press, Xi'an (1990)
11. McClelland, J.L., Rumelhart, D.E.: Explorations in Parallel Distributed Processing. In: A handbook of Model, Programs and Exercises. MIT Press, Cambridge (1986)
12. Martin, T.H., Howard, B.D., Mark, B.: Neural Network Design. PWS Publishing Company (1996)
13. Moody, J., Darken, C.: Fast learning in networks of locally-tuned processing units. Neural Computation 1, 281–294 (1989)
14. Yue, C.Y.: Decision-making theory and Method. Science Press, Beijing (2003)
15. Li, M.Q., Kou, J.S.: Basic theory and application of genetic algorithm. Science Press, Beijing (2002)

Connectivity Control Methods and Decision Algorithms Using Neural Network in Decentralized Networks

Demin Li[1,*], Jie Zhou[2], Jiacun Wang[3], and Chunjie Chen[1]

[1] College of Information Science and Technology, Donghua University, Songjiang District, Shanghai 201620, China
deminli@dhu.edu.cn
[2] School of Science, Donghua University, Songjiang District, Shanghai 201620, China
[3] Department of Computer Science, Monmouth University, West Long Branch, NJ 07762, USA

Abstract. Consider mobile agents' power limitation and mobility resulting in communication delay, network connectivity may be not maintained only by power control or mobility control in mobile decentralized network, so we provide integrated power and mobility control law design methods for preserved connectivity in this paper. Those integrated connectivity control methods are achieved by constructing decentralized navigation functions respective to relative position constrain conditions. We extend previous works of power control and mobility control for connectivity, and design power control methods, integrated power and mobility control methods for position or velocity. Furthermore, considering power control may have the same effect with mobility control on connected distance control, we present some decision algorithms for decentralized networks connectivity using neural networks. Numerical simulations are discussed in the end. The results could be applied to location management, data consensus and decision making in decentralized networks.

Keywords: preserved connectivity, integrated control, time delay, navigation functions, decision algorithm.

1 Introduction

Mobile ad hoc networks (MANET) are mobile and wireless sensor networks. Not only can the networks collect information, just like neural networks, but also the mobile nodes in the networks,just as neuron, can communication each other. The integrated study on networks and control is a very challenging and promising research area [1,2]. Since mobile sensor nodes or mobile agents in decentralized networks may spread in an arbitrary manner, one of the fundamental issues for location management, communication and data consensus in a mobile decentralized network is to maintain connectivity. Connectivity control may be classified as mobility control (see[3,4,5] and references there in) and power or topology control (see[6,7,8] and references there in). Increasing power only for communication may shorten the lifetime of networks; mobility control only may enlarger the communication delay.

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 152–160, 2010.

To mobility control for connectivity, Michael M. Zavlanos and George J. Pappas [3] consider graph connectivity or mobility control as a primary objective, and propose a distributed feedback control framework based on novel control decomposition. Communication time delays in the networks as well as collision avoidance among the agents can also be handled. In paper [4], a feedback control strategy that achieves convergence of a multi-agent system to desired formation connectivity is proposed for both the cases of agents with single integrator and nonholonomic unicycle-type kinematics. The stability of distance-based formation control is discussed in [5]. In particular, the authors proposed formation control law that is based on the negative gradient of a potential function between each of the pairs of agents that form an edge in the formation graph.

To power control for connectivity, Jie Wu, and Fei Dai [4] showed some issues in existing topology control, and propose a mobility-sensitive topology control method that extends many existing mobility-insensitive protocols. Two mechanisms are introduced: consistent local views that avoid inconsistent information and delay, and mobility management that tolerate outdated information. The effectiveness of the proposed approach is confirmed through an extensive simulation study. Javier Gomez, and T. Andrew [7] showed that there is an optimum setting for the transmission range, not necessarily the minimum, which maximizes the capacity available to nodes in the presence of node mobility. The results presented in this paper highlight the need to design future MAC and routing protocols for wireless ad hoc and sensor networks based, not on common-range which is prevalent today, but on variable-range power control. P. Siripongwutikorn, B. Thipakorn [8] proposed two topology control algorithms – ABsolute Distance-based (ABD) and Predictive Distance-based (PRD), which adjust the transmission range of individual nodes in a MANET to achieve good network throughput, particularly under correlated node movements as in a vehicular environment. Both algorithms attempt to maintain the number of logical neighbours between two predefined thresholds. To best to our knowledge, we can not find out any results for integrated power and mobility control law design for connectivity.

It is now well recognized that the dynamical behaviors of many communication processes contains inherent time delays. Time delays may result from the distributed nature of the communication system, information transitions, or from the time required to measure some of the signals. It is known that processes with time delays are inherently difficult to analyze and control [9], in the sense that it is difficult to achieve satisfactory performance. Therefore, analysis and design of time-delay systems with various types of disturbances has been a subject of great practical importance for several decades, see [10-12] and the references therein. Unfortunately, so far, the integrated power and mobility control methods for preserved connectivity in decentralized networks with communication delay has yet to be investigated, which helps motivate our current study.

In this paper, inspired by previous works in [4,5], we deal with the integrated power and mobility control methods for preserved connectivity in decentralized networks. An effective navigation function and Lyapunov function approach are extended and developed to solve the preserved connectivity with time delay problem. Furthermore, considering power control may have the same effect with mobility control on connected distance control, we present some decision algorithms for decentralized networks connectivity using neural networks. A numerical example is presented to show the effectiveness and efficiency of the proposed control law design scheme.

The remainder of the paper is arranged as follows. The definition of preserved connectivity is formulated and power control law design is discussed in section 2 for decentralized networks with no movement nodes. In section 3, Consider mobile agents' power limitation and mobility, preserved connectivity may be not maintained only by power control in mobile decentralized network, so integrated power and mobility control law design is provided. In section 4, considering power control may have the same effect with mobility control on connected distance control, we present decision algorithms for connectivity using neural network. Numerical simulations are discussed in section 5. Finally, some concluding remarks are given in the end.

2 Power Control Law Design for Preserved Connectivity

The communication graph $G = \{V, E\}$ is an undirected graph consisting of a set of vertices $V = \{1,2,...,N\}$ indexed by the mobile agents and a set of edges $E = \{(i, j) | i, j \in V\}$. $p_i \in R^2$ denotes the position of mobile agent i .Each i 's objective is to converge to a particular relative position vector $c_{ij} = p_i - p_j \in R^2$,and $\|c_{ij}\| = \sqrt{(p_{ix} - p_{jx})^2 + (p_{iy} - p_{jy})^2}$ is relative distance of node i to node j . By the relative position vectors, the mobile decentralized network is connected.

Definition 1 (Preserved Connectivity). The given relative position vector set $C = \{c_{ij} | 0 < \|c_{ij}\| < d, i, j = 1,2,..., N\}$ in the communication graph $G = \{V, E\}$ is a preserved connectivity, if the G is connected for the given position vector set C .Where d is the maximum sensing radius of every mobile agent.

Mobile agent' power is always related to its communication radius $r_i(t)$, the larger its communication radius is, and the bigger its power is required. So, we manage to design control laws for agents' communication radius to save its energy while the preserved connectivity is maintained. We consider N single integrator point mobile agents in the plane.

$$\dot{r}_i(t) = u_i(t) \quad (i = 1,2,..., N) \tag{1}$$

where the $u_i(t)$ is control input for each mobile agent i .

The power of each agent is equipped with a navigation function

$$H_i = \frac{1}{2}[(d - c_i)^2 - (r_i(t) - c_i)^2]$$

where $N_i = \{j | c_{ij} \in C, 0 < \|c_{ij}\| < d\}$, $c_i = \max_{1 \le j \le N_i} \{\|c_{ij}\|\}$, $c_i < r_i(t) < d$, and d is the maximum sensing radius of every mobile agent.

Remark 1. Where the N_i stands for the neighborhoods of the mobile node i , and in other words, the $|N_i|$ is the number of one-hop communication nodes of the node i .

We can see from the navigation function H_i that with the $r_i(t)$ decreasing, the H_i will be increasing. In other words, we can control the power of node i by H_i.

So the power control law of each mobile agent i is now defined by

$$u_i(t) = \frac{\partial H_i}{\partial r_i} = -(r_i(t) - c_i) \tag{2}$$

substitute the $u_i(t)$ in (1), we have

$$\dot{r}_i(t) = -(r_i(t) - c_i) \tag{3}$$

The conclusion is summarized in the following theorem

Theorem 1. Assume that (1) is driven by (2), the radius $r_i(t)$ of mobile agent i in system (3) converges to the desired $c_i = \max_{1 \le j \le N_i} \{\|c_{ij}\|\}$.

Proof from the (3), we easily get

$$\frac{d[r_i(t) - c_i]}{dt} = -(r_i(t) - c_i)$$

then $\lim_{t \to +\infty} r_i(t) = c_i$ .The theorem is proved.

3 Integrated Power and Mobility Control Law Design

In section 2, we discuss the power control laws design for preserved connectivity. Consider mobile agents' power limitation and mobility constrains, preserved connectivity may be not maintained only by power or mobility for mobile decentralized network, so we describe integrated power and mobility control laws design.

$$\begin{aligned} \dot{r}_i(t) &= u_i(t) \\ \dot{p}_i(t) &= v_i(t) \end{aligned} \tag{4}$$

Similar section 2 to construct the power navigation function, we extend navigation function described in [4], and present an integrated power and mobility navigation function for system (4)

$$G_i(t) = \frac{1}{2}(r_i(t) - c_i)^2 + \frac{1}{2}\sum_{j \in N_i} \|p_i(t) - p_j(t) - c_{ij}\|^2$$

and $N_i = \{j | c_{ij} \in C, 0 < \|c_{ij}\| < d\}$, $c_i = \max_{1 \le j \le N_i} \{\|c_{ij}\|\}$, $c_i < r_i(t) < d$

Remark 2. We can see from the navigation function $G_i(t)$ that with the $r_i(t)$ decreasing, and with relative positions approaching to the preserved positions, the $G_i(t)$ will be increasing. In other words, we can control the power and mobility of

node i by $G_i(t)$ .So, for preserved connectivity set C , to maintain the desired connectivity and enlarge the lifetime of decentralized networks, integrated power and mobility navigation function $G_i(t)$ to navigate $r_i(t)$ approaching to c_i , and to navigate $p_i(t) - p_j(t)$ approaching to c_{ij} .

The power control law of each mobile agent i is now defined by

$$u_i(t) = -\frac{\partial G_i(t)}{\partial r_i(t)} \tag{5}$$

and mobility control law of each mobile agent i is defined by

$$v_i(t) = -\frac{\partial G_i(t)}{\partial p_i(t)} \tag{6}$$

we get closed loop power control system from (4)(5)

$$\dot{r}_i(t) = -(r_i(t) - c_i) \quad (3)$$

and closed loop mobility control system from (4)(6)

$$\dot{p}_i(t) = -\sum_{j \in N_i} [p_i(t) - p_j(t) - c_{ij}] \tag{7}$$

The conclusion is summarized in the following theorem

Theorem 2. Assume the power of (4) is driven by (5), and the mobility of (4) is driven by (6), $c_i < r_i(t) < d$, $c_i = \max_{1 \le j \le N_i} \{\|c_{ij}\|\}$, then $r_i(t)$ converges to c_i $(i = 1,2,...,N)$, the relative position of mobile agent i and j , $p_{ij}(t) = p_i(t) - p_j(t)$ converges to the desired formation connectivity c_{ij} $(i, j = 1,2,...,N)$.

Proof. Let $\hat{r}_i(t) = r_i(t) - c_i$ in (3) and $\hat{r}(t) = (\hat{r}_1(t), \hat{r}_2(t),..., \hat{r}_N(t))^T$, the (3) can now be written in stack vector form as

$$\frac{d\hat{r}(t)}{dt} = -I_N \hat{r}(t) \tag{8}$$

Where I_N is $N \times N$ identity matrix. Similarly, from the (7), let $c_{ij} = \bar{c}_i - \bar{c}_j$,we easily get

$$\frac{d[p_i(t) - \bar{c}_i]}{dt} = -\sum_{j \in N_i} [(p_i(t) - \bar{c}_i) - (p_j(t) - \bar{c}_j)]$$

and let $\hat{p}_i(t) = p_i(t) - \bar{c}_i$, then we deduce

$$\frac{d\hat{p}_i(t)}{dt} = -\sum_{j \in N_i} [\hat{p}_i(t) - \hat{p}_j(t)] \tag{9}$$

Let $\hat{p}(t) = (\hat{p}_1^T(t), \hat{p}_2^T(t),..., \hat{p}_N^T(t))^T$

the (7) can also now be written in stack vector form as

$$\frac{d\hat{p}(t)}{dt} = -[P \otimes I_2]\hat{p} \tag{10}$$

where the matrix P is defined as

$$P_{ij} = \begin{cases} |N_i|, i = j \\ -1, j \in N_i \\ 0, j \notin N_i \end{cases}$$

Where $|N_i|$ is the number of element in set N_i, the matrix P has zero row sums, is the Laplacian matrix of the connectivity graph subject to c_{ij} ($i, j = 1,2,...,N$) , $\otimes$ is Kronecker product, and consider $\hat{p}_i(t) \in R^2$, I_2 is 2×2 identity matrix.

$$\text{Let } q(t) = (\hat{r}^T(t), \hat{p}^T(t))^T$$

We examine the stability of the closed loop system by using the candidate Lyapunov function

$$V(q) = \sum_{i=1}^{N} G_i(q) > 0$$

Its gradient can be computed as

$$\dot{V}(q) = [\frac{\partial V}{\partial \hat{p}}]^T \frac{d\hat{p}}{dt} + [\frac{\partial V}{\partial \hat{r}}]^T \frac{d\hat{r}}{dt} = -2\left\|[P \otimes I_2]\hat{p}\right\|^2 - \left\|I_N \hat{r}\right\|^2 < 0$$

so we conclude that $\lim_{t \to +\infty} q(t) = 0$,the theorem is proved.

4 Decision Algorithms for Connectivity Using Neural Network

We see from the above discussions that power control and mobility control are not considered contemporarily, only discussed individually. We know that power control may have the same effect with mobility control on connected distance control. For example, the node A and the node B can not communication each other because the distance between two nodes is larger than the transmit radius of the node A or the node B . If the two nodes want to communicate each other, there are some policies to be operated: (1) the node A can move to the node B ;(2) the node A can increase its power; (3) the node B can move to the node A ; (4) the node B can increase its power; (5) the node A and the node B can move simultaneously to decrease the distance between them; (6) the node A and the node B can increase its power simultaneously to get connectivity between them; (7) one node use power control, the other use mobility control; etc. B. P. Vijay Kumar and P. Venkataram [14] proposed connectivity management technique (CMT) is to maintain the mobile user-to-user connectivity

throughout the transaction in centralized networks. The CMT keeps track of the interrupted and hand-off mobile hosts (MHs) by maintaining the status information at the currently registered base station (BS). The status information contains the activities of all the MHs registered with the BS. The CMT uses a neural network that is trained with respect to the status information, both online and offline training, to learn the volatile nature of the mobile environment for an intelligent decision. We highlight from this method and consider the decentralized networks properties, we present some decision algorithms for decentralized networks connectivity using neural networks. We adopt the neural network training algorithm in [13,14], the decentralized networks connectivity decision algorithm is described briefly.

Suppose every node maintains real time its connectivity status information based on the messages exchanged among its neighbor nodes. The connectivity status information in one mobile node includes the information of the relative position of its neighbor nodes. If a mobile node can not transmit to the next nodes, it knows that the network is disconnected, and must operate mobility or power control decision policies to achieve the links among its neighbors. The node connectivity decision algorithm processes the recent connectivity status information in its cache and derives the training data set in the mobile node. The neural network is trained on the training data set which is derived from connectivity status information using back-propagation learning algorithm, and the decision of the mobile node on how to operate is decided.

Begin
1. Every node maintains real time its connectivity status information based on the messages exchanged among its neighbor nodes;
2. Training data set is derived from its connectivity status information;
3. Neural network is trained on the training data set using back-propagation learning algorithm;
4. Once a mobile node can not transmit to the next nodes, the every node connectivity decision algorithm processes the recent connectivity status information in its cache and derives the training data set in the mobile node;

then
go to step 3;
else
Output the decision of the mobile node on how to operate in the next step;
go to step 4.
Stop
End.

Fig. 1. Decision algorithms for connectivity using neural network

5 Numerical Simulations

We propose numerical simulations for the integrated power and mobility control law design with time constant delay.

Let $p_1(0)=(200,0)$, $p_2(0)=(0,300)$, $p_3(0)=(200,550)$, $p_4(0)=(400,550)$ and $p_5(0)=(575,280)$ be the initial position of five nodes. Consider $d=100\sqrt{13}$, $t\in[0,25]$ we have adjacent matrix A of the connectivity graph and the Laplacian matrix P of A

$$A=\begin{pmatrix}0&1&0&0&0\\1&0&1&0&0\\0&1&0&1&0\\0&0&1&0&1\\0&0&0&1&0\end{pmatrix}\quad P=\begin{pmatrix}1&-1&0&0&0\\-1&2&-1&0&0\\0&-1&2&-1&0\\0&0&-1&2&-1\\0&0&0&-1&1\end{pmatrix}$$

We get the final position of five nodes by control law in (16) $p_1(25)=(300,250)$, $p_2(25)=(120,100)$, $p_3(25)=(325,400)$, $p_4(25)=(550,400)$, and $p_5(25)=(400,75)$. The connectivity graph of these five nodes at $t=25$ is shown in Fig. 2. We can see the five nodes are not connected at $t=0$ (the connectivity graph is drawn with real blue line), and that the five nodes are connected at $t=25$ (the connectivity graph is drawn with dash read line).The green circles in Fig.2 stand for the dynamic moving process at $t\in[0,25]$ with control law (16).

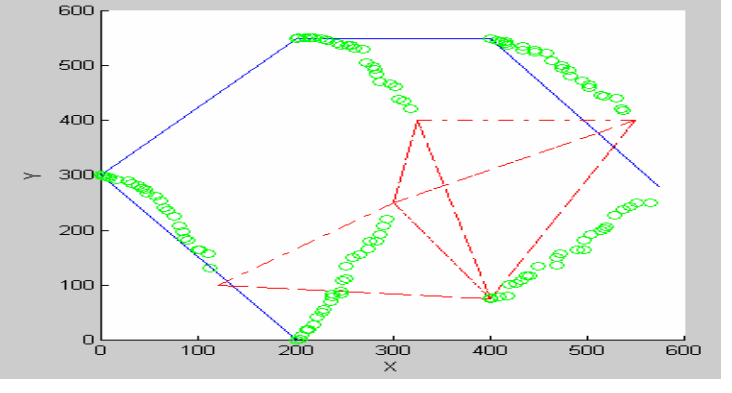


Fig. 2. The connectivity graph of five nodes at $t=25$

6 Conclusion

We intensively discuss the integrated power and mobility control laws design methods in decentralized networks. With time increases, the communication radius approach to the preserved value; the relative positions converge or synchronize to the preserved position; the velocities converge to a constant. Considering power control may have the same effect with mobility control on connected distance control, we present some decision algorithms for decentralized networks connectivity using neural networks.

Preserved connectivity may relate to mobile models of agents, such as random waypoint model, random walk model, and random direction model, etc. We are managing to deal with those challenges.

Acknowledgement

This work is partially supported by Shanghai Key Basic Research Project under grant number 09JC1400700; Shanghai Fund of Science and Technology granted number 00JG05047; NSFC granted number 70271001; China Postdoctoral Fund granted number 2002032191; Shanghai Key Scientific Research Project under grant number 05dz05036; and 2008 Fund of Engineering Research Centre of Digitized Textile & Fashion Technology, Ministry of Education.

References

1. Yang, F., Wang, Z., Hung, Y.S., Gani, M.: H-infinity control for networked systems with random communication delays. IEEE Transactions on Automatic Control 51(3), 511–518 (2006)
2. Lin, C., Wang, Z., Yang, F.: Observer-based networked control for continuous-time systems with random sensor delays. Automatica 45(2), 578–584 (2009)
3. Zavlanos, M.M., Pappas, G.J.: Distributed Connectivity Control of Mobile Networks. In: Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, December 12-14 (2007)
4. Dimarogonas, D.V., Kyriakopoulos, K.J.: A connection between formation infeasibility and velocity alignment in kinematic multi-agent systems. Automatica 44, 2648–2654 (2008)
5. Dimarogonas, D.V., Kyriakopoulos, K.J.: On the stability of distance-based formation control. In: Proceedings of the 47th IEEE Conference on Decision and Control, Cancun, Mexico, December 9-11, pp. 1200–1205 (2008)
6. Wu, J., Dai, F.: Mobility-Sensitive Topology Control in Mobile Decentralized Networks. IEEE Transactions on Parallel and Distributed Systems 17(6), 522–535 (2006)
7. Gomez, J., Campbell, A.T.: Variable-Range Transmission Power Control in Wireless Decentralized Networks. IEEE Transactions on Mobile Computing 6(1), 87–99 (2007)
8. Siripongwutikorn, P., Thipakorn, B.: Mobility-aware topology control in mobile decentralized networks. Computer Communications 31, 3521–3532 (2008)
9. Gorecki, H., Fuksa, S., Grabowski, P.: Analysis and synthesis of time delay systems. Wiley, New York (1989)
10. Gu, G., Lee, E.B.: Stability testing of time delay systems. Automatica 25, 777–780 (1989)
11. Chen, J., Gu, G., Nett, C.N.: A new method for computing delay margins for stability of linear delay systems. Syst. and Contr. Lett. 26, 107–117 (1995)
12. Li, D., Wang, Z., Zhou, J., Fang, J., Ni, J.: A note on chaotic synchronization of time-delay secure communication systems. Chaos, Solitons and Fractals 38, 1217–1224 (2008)
13. Cao, J., Tao, Q.: Estimation of the Domain of Attraction and the Convergence Rate of a Hopfield Associative Memory and an Application. J. Comput. Syst. Sci. 60(1), 179–186 (2000)
14. Vijay Kumar, B.P., Venkataram, P.: A Neural Network–Based Connectivity Management for Mobile Computing Environment. International Journal of Wireless Information Networks 10(2), 63–71 (2003)

A Quantum-Inspired Artificial Immune System for Multiobjective 0-1 Knapsack Problems

Jiaquan Gao, Lei Fang, and Guixia He

Zhijiang College, Zhejiang University of Technology, Hangzhou 310024, China
gaojiaquan@gmail.com

Abstract. In this study, a novel quantum-inspired artificial immune system (MOQAIS) is presented for solving the multiobjective 0-1 knapsack problem (MKP). The proposed algorithm is composed of a quantum-inspired artificial immune algorithm (QAIS) and an artificial immune system based on binary encoding (BAIS). On one hand, QAIS, based on Q-bit representation, is responsible for exploration of the search space by using clone, mutation with a chaos-based rotation gate, update operator of Q-gate. On the other hand, BAIS is applied for exploitation of the search space with clone, a reverse mutation. Most importantly, two diversity schemes, suppression algorithm and truncation algorithm with similar individuals (TASI), are employed to preserve the diversity of the population, and a new selection scheme based on TASI is proposed to create the new population. Simulation results show that MOQAIS is better than two quantum-inspired evolutionary algorithms and a weight-based multiobjective artificial immune system.

Keywords: multiobjective, knapsack problem, artificial immune system, quantum computing.

1 Introduction

The multiobjective 0-1 knapsack problem (MKP), as a NP-hard, classic multiobjective optimization problem (MOOP), is an extension of the single objective version of this problem. For MKP, it always plays a important role in our real lives and has been frequently used to examine the performance of evolutionary multiobjective algorithms in the literature [1,2].

In this paper, MKP will continue to be studied and a novel quantum-inspired artificial immune system (MOQAIS) based on artificial immune system (AIS) and quantum computing is presented and has the following characteristics. First, two algorithms with different representations are combined, QAIS uses quantum-bit (Q-bit) representation, and BAIS adopts binary representation. Second, different mutation is combined, where QAIS utilizes quantum-gate (Q-gate) rotation, and BAIS utilizes reverse mutation. Thirdly, exploration and exploitation are considered simultaneously, where QAIS emphasizes exploration, and BAIS emphasizes exploitation. Finally, QAIS and BAIS both apply similarity-based selection method to generate the new population, which guarantees the diversity of

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 161–168, 2010.

the population. Numerical results on several test problems show MOQAIS outperforms the quantum-inspired multiobjective evolutionary algorithm (QMEA) [3], a hybrid quantum genetic algorithm (HQGA) [4] and a weight-based multiobjective artificial immune system (WBMOAIS) [5].

2 MOQAIS

In this section, a novel quantum-inspired AIS (MOQAIS) is presented. MOQAIS is an integrated framework of two algorithms QAIS and BAIS. Before proposing MOQAIS, assume that the population is a set of cells and each of them represents a solution to MKP, the main procedures of QAIS and BAIS are respectively listed in the following.

1. The main procedures of QAIS:

(1) A random initial population P_0^{Q} of size $N_{\mathrm{pop}}^{\mathrm{Q}}$ is created and set the memory $M^{\mathrm{Q}} = \emptyset$. Repair population P_0^{Q} and set a counter $t = 0$.
(2) For each cell in the population, reproduce the cell $N_{\mathrm{clones}}^{\mathrm{Q}}$ times and mutate each clone by a chaos-based rotation gate.
(3) Repair the offspring, and then combine the population P_t^{Q} and the offspring as R_t^{Q}.
(4) Evaluate R_t^{Q}, and perform trimming operation for R_t^{Q}. Compute all nondominated cells in R_t^{Q}, and store them to M^{Q}, and then delete all dominated cells in M^{Q}.
(5) Select new population P_{t+1}^{Q} from R_t^{Q} by using the selection scheme based on truncation algorithm with similar individuals (TASI).
(6) Randomly select a cell from M^{Q}, and use it to update the new population P_{t+1}^{Q} by a rotation gate.
(7) if $t > T^{\mathrm{Q}}$ (the maximum number of generations) or another termination condition is satisfied, then terminate the algorithm and output the nondominated cells in M^{Q}. Otherwise, let $t = t + 1$ and return to (2).

2. The main procedures of BAIS:

(1) Let t=0 and initialize the population P_0^{B} of size $N_{\mathrm{pop}}^{\mathrm{B}}$ together with the population obtained by QAIS. Set the memory $M_0^{\mathrm{B}} = \emptyset$ and its maximum size $= N_{\mathrm{mem}}$.
(2) For each cell in the population, reproduce the cell $N_{\mathrm{clones}}^{\mathrm{B}}$ times and mutate each clone by a reverse mutation operator.
(3) Repair the offspring, and then combine the population P_t^{B}, the memory M_t^{B} and the offspring as R_t^{B}.
(4) Evaluate R_t^{B}, and perform trimming operation for R_t^{B}.
(5) Compute nondominated cells among R_t^{B}, and copy them to the memory M_{t+1}^{B}.
(6) if $t > T^{\mathrm{B}}$ (the maximum number of generations) or another termination condition is satisfied, then terminate the algorithm and output the nondominated cells in the memory. Otherwise, go on the following step.

(6) If $|M_{t+1}^{\mathrm{B}}| > N_{\mathrm{mem}}$, then perform TASI to reduce the size of M_{t+1}^{B} to N_{mem}.
(7) If $|M_{t+1}^{\mathrm{B}}| > N_{\mathrm{pop}}^{\mathrm{B}}$, copy cells of M_{t+1}^{B} to a set Z_{t+1}^{B}, and use TASI to reduce the size of Z_{t+1}^{B} to $N_{\mathrm{pop}}^{\mathrm{B}}$, and then let $P_{t+1}^{\mathrm{B}} = Z_{t+1}^{\mathrm{B}}$. Otherwise, copy cells of M_{t+1}^{B} to P_{t+1}^{B}, and then select $(N_{\mathrm{pop}}^{\mathrm{B}} - |M_{t+1}^{\mathrm{B}}|)$ cells from dominated cells among R_t^{B} by using the selection scheme based on TASI, and add them to P_{t+1}^{B}. Let $t = t + 1$, and go to (2).

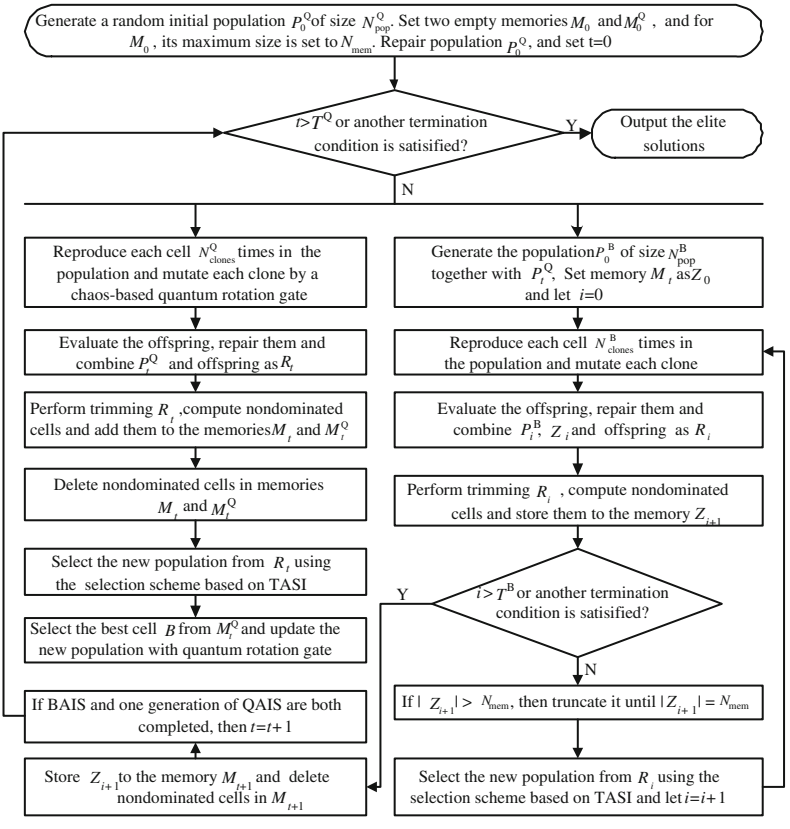


Fig. 1. The main framework of MOQAIS

On the basis of QAIS and BAIS, the main framework of MOQAIS is shown in Fig. 1. From Fig. 1, It can be seen that the quantum-inspired search and immune search are hybridized, and the proposed algorithm is structured into two nested levels. The inner level manages the exploitation of the search space by using BAIS. The task of exploration of the solution space is carried out in the outer level by using QAIS. For the trimming scheme and the selection scheme, we refer the reader to the literature [6].

2.1 Encoding and Decoding Technique

For QAIS, the encoding technique based on Q-bit is adopted. A cell is represented as a string of n Q-bits in the following:

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \beta_1 & \beta_2 & \cdots & \beta_n \end{array}\right]$$

where $|\alpha_i|^2 + |\beta_i|^2 = 1$, $i = 1, 2, \ldots, n$. For the Q-bit string, it can be obviously seen that it can not be directly used to calculate the objective values. Thus, a random key decoding technique is applied. According to [7], a Q-bit string is first converted to a binary string. In particular, for the jth ($j = 1, 2, \cdots, n$) bit of Q-bit string, a random number η is generated at interval [0,1]. If $|\beta_j|^2 > \eta$, then let the corresponding bit r_j of the binary string be 1; otherwise let r_j be 0. In BAIS, a binary encoding is directly applied, and the objective values can be directly obtained using (4).

2.2 Mutation Operator

In our proposed algorithm, two distinct mutation operators are applied to BAIS and QAIS, respectively. In BAIS, a reverse mutation operator is used to perform mutation for each cell. And for a cell, every gene is mutated with a predefined probability p_{m}.

For example, for a nine-item MKP, a parent cell is [0 1 0 1 1 0 0 1 1]. If the second, sixth, and eighth genes are chosen to perform mutation, then the children will be [0 0 0 1 1 1 0 0 1].

For QAIS, a chaos rotation Q-gate is utilized to perform mutation for each cell. Like BAIS, every Q-bit is mutated with a predefined probability p_{m}. Assume that a parent cell is $\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \beta_1 & \beta_2 & \cdots & \beta_n \end{array}\right]$, its mutation procedure is listed as follows.

(1) $j = 1$
(2) Generate a random number η at interval $[0, 1]$.
(3) If $\eta < p_{\mathrm{m}}$, (α_j, β_j) of the jth Q-bit is updated as follows.

$$\begin{bmatrix} \alpha_j' \\ \beta_j' \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta_j) & -\sin(\Delta\theta_j) \\ \sin(\Delta\theta_j) & \cos(\Delta\theta_j) \end{bmatrix} \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} \tag{1}$$

where $\Delta\theta_j$ is the angle parameter of rotation Q-gate, and $\Delta\theta_j = \lambda x_k^j$. x_k^j is a chaos variable, and be calculated by the following equation:

$$x_k^j = 8x_{k-1}^j(1 - x_{k-1}^j) - 1. \tag{2}$$

From (2), it can be observed that $\Delta\theta_j$ is within the range $[-\lambda, \lambda]$.
(4) $j = j + 1$.
(5) If $j > n$, then terminate the procedure. Otherwise, return to (2).

In the above procedure, λ is an amplitude, and its value depends on the rank of the parent cell in the population. Assume that the number of nondominated fronts in the population is K, for a parent cell, if it is on the ith front ($i = 1, 2, \cdots, K$),

$$\lambda = \lambda_0 \exp((i - |K|)/|K|), \tag{3}$$

where λ_0 is a control parameter. From (1)-(3), we can see that for a parent cell, the better its rank is, the smaller the chaos disturb is. And vice versa.

2.3 Rotation Gate for Q-Bit in QAIS

The rotation gate is only used in QAIS to adjust the probability amplitudes of each Q-bit. According to the rotation gate in (1), a quantum-gate $U(\theta_i)$ is a function of $\theta_i = s(\alpha_i, \beta_i) \cdot \Delta\theta_i$, where $s(\alpha_i, \beta_i)$ is the sign of θ_i that determines the direction and $\Delta\theta_i$ is the magnitude of rotation angle. Like the literature [4,7], the angle parameters used for the rotation gate are selected from the lookup table (see Table 1). In Table 1, b_i and r_i are the ith bits of a nondominated solution b randomly selected from the memory and a binary solution r, respectively. Because b comes from the nondominated set in QAIS, the use of Q-gate is to emphasize the searching direction toward b, which will be helpful for enhancing the quality of the population.

Table 1. Lookup table of rotation angle

r_i	b_i	$f(r) < f(b)$	$\Delta\theta_i$	$s(\alpha_i, \beta_i)$			
				$\alpha_i\beta_i > 0$	$\alpha_i\beta_i < 0$	$\alpha_i = 0$	$\beta_i = 0$
0	0	true	0	0	0	0	0
0	0	false	0	0	0	0	0
0	1	true	0.05π	+1	-1	0	± 1
0	1	false	0	0	0	0	0
1	0	true	0.01π	-1	+1	± 1	0
1	0	false	0.025π	+1	-1	0	± 1
1	1	true	0.005π	+1	-1	0	± 1
1	1	false	0.025π	+1	-1	0	± 1

2.4 Repairing Scheme

In our proposed algorithm, some unfeasible solutions will be generated at each reproduction generation, which will influence the performance of algorithm. Thus, the following greedy repairing approach is utilized to transform the unfeasible solutions to feasible solutions. First, compute the maximum profit/weight ratio $q_j = \max_{i=1}^{m}\{p_{ij}/w_{ij}\}$, $j = 1, 2, \cdots, n$. Second, sort q_j in the decreasing order. Finally, the selection procedure always chooses the last item for deletion until the constrained conditions are satisfied.

3 Experiments

3.1 Test Problem

MKP can be formulated as follows [1]:

$$\text{Maximize } f_i(\mathbf{x}) = \sum_{j=1}^{n} p_{ij} x_j, \quad i = 1, 2, \ldots, m \tag{4}$$

subject to

$$\sum_{j=1}^{n} w_{ij} x_j \leqslant c_i, \quad \text{for } \forall i \in \{1, 2, \ldots, m\} \tag{5}$$

where each f_i, $1 \leqslant i \leqslant m$, is an objective function, $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T \in \{0,1\}^n$, p_{ij} is the profit of item j according to knapsack i, w_{ij} is the weight of item j according to knapsack i, and c_i is the capacity of knapsack i. If $x_j = 1$, the jth item is selected for the knapsack.

Test data come from the following two aspects. On one hand, uncorrelated profits and weights are considered. w_{ij} and p_{ij} are randomly generated with an uniform distribution at interval [10,30], and the knapsack capacities are set to half the total weight. For this case, the knapsack problems, (2,750) and (3,250) are considered. On the other hand, we use strongly correlated sets of data. For the first knapsack, w_{ij} is randomly generated with a uniform distribution at interval [1,100], and $p_{1j} = w_{1j} + 30$; for the other knapsacks, w_{ij} and p_{ij} are randomly generated with an uniform distribution at interval [1,100]. Like the above data, the average knapsack are used. In this case, the three knapsack problems, (2,100), (2,250) and (3,500) are considered.

3.2 Performance Measures

(1) Diversity metric(D): In order to measure how well the solutions throughout nondominated set (Denoted by Q) are distributed, the metric proposed by Li et al. [8] is used to evaluate the spread of the set Q of nondominated solutions, and can be formulated as:

$$D = \frac{\sum_{i=1}^{m} (f_{\max}^i - f_{\min}^i)}{\sqrt{\frac{1}{|Q|} \sum_{i=1}^{|Q|} (d_i - \bar{d})^2}}, \tag{6}$$

where d_i is the Euclidean distance between the ith solution and its closest neighbor, $\bar{d}$ is the mean distance, and $f_{\max}^i$ ($f_{\min}^i$) denote the maximum (minimum) value of the ith objective. The greater value of the diversity metric, the better the diversity of nondominated solutions is.

(2) Coverage of two sets (C): In order to measure the convergence of nondominated solutions, the metric suggested by Zitzler [1] is used. For two sets of solution vectors A and B, the set coverage metric $C(A, B)$ calculates the proportion of solutions in B, which are weakly dominated by solutions of A:

$$C(A, B) = \frac{|\{b \in B | \exists a \in A : a \preceq b\}|}{|B|}. \tag{7}$$

Table 2. Termination conditions of all algorithms

Problems	(2,100)	(2,250)	(2,750)	(3,250)	(3,500)
Fitness function evaluations	115,000	225,000	450,000	150,000	150,000

Table 3. D comparision

Problems	MOQAIS		HQGA		QMEA		WBMOAIS	
	MV	SD	MV	SD	MV	SD	MV	SD
(2,100)	5.224999	1.118759	2.623325	0.871489	1.257989	0.586969	2.128839	0.588696
(2,250)	9.984834	1.922662	7.012672	2.201354	3.142502	1.041164	5.275780	1.195882
(2,750)	23.21331	3.458199	7.969381	2.391153	4.336546	1.349201	9.882045	2.900051
(3,250)	11.44253	1.624752	10.98980	2.128796	7.626542	0.608593	9.875016	0.871207
(3,500)	15.83841	3.013895	8.301092	3.895801	2.232156	1.975013	3.168448	3.789541

Table 4. C comparision

	Problems	MOQAIS(B)	HQGA(B)	QMEA(B)	WBMOAIS(B)
MOQAIS(A)	(2,100)	/	0.343	1	0.894
	(2,250)	/	0.966	1	0.960
	(2,750)	/	1	1	1
	(3,250)	/	0.815	1	0.743
	(3,500)	/	0.453	1	0.897
HQGA(A)	(2,100)	0.478	/	1	0.894
	(2,250)	0.028	/	1	0.548
	(2,750)	0	/	1	0
	(3,250)	0.043	/	1	0.412
	(3,500)	0.200	/	1	0.654
QMEA(A)	(2,100)	0	0	/	0
	(2,250)	0	0	/	0
	(2,750)	0	0	/	0
	(3,250)	0	0	/	0
	(3,500)	0	0	/	0
WBMOAIS(A)	(2,100)	0.048	0.035	1	/
	(2,250)	0.007	0.248	1	/
	(2,750)	0	0.969	1	/
	(3,250)	0.232	0.689	1	/
	(3,500)	0.112	0.235	1	/

The metric value $C(A, B) = 1$ means all members of B are weakly dominated by A. On the other hand, $C(A, B) = 0$ means that no member of B is weakly dominated by A.

3.3 Computation Results

The proposed algorithm MOQAIS is compared against two quantum-inspired algorithms: QMEA [3] and HQGA [4]. In addition, MOQAIS is also compared

with a multiobjective artificial immune system WBMOAIS [5] in order to verify the quantum-inspired effect. In this case, the termination conditions are listed in Table 2 for all four algorithms. The values of the two metrics for all algorithms are respectively presented in Tables 3 and 4. In Table 4, MV and SD mean the mean value and the standard deviation, respectively, and MV = the obtained mean value/10, and SD=the obtained standard deviation/10.

By analyzing the numerical results presented in Tables 3 and 4, our observation is confirmed. We can see that for the problems which include (2,250), (2,750), (3,250) and (3,500), MOQAIS obtains a better value than HQGA, QMEA and WBMOAIS for each metric. For the problem (2,100), C(MOQAIS, HQGA)=0.343 and C(HQGA, MOQAIS)=0.478, so we can see that HQGA shows a slightly better behavior than MOQAIS for the metric C. However, for the other three metrics, MOQAIS has the best values. Therefore, we can affirm that MOQAIS is better than HQGA, QMEA and WBMOAIS.

4 Conclusion

In this study, we present a novel quantum-inspired multiobjective artificial immune system for MKP. The experimental results for MKP with 5 different test data support the claim that the proposed algorithm exhibits better proximity performance as well as diversity maintenance, and outperforms two other quantum-inspired evolutionary algorithms, QMEA and HQGA, and a weight-based multiobjective artificial immune system, WBMOAIS.

References

1. Zitzler, E., Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Trans. Evol. Comput. 3(4), 257–271 (1999)
2. Ishibuchi, H., Narukawa, K., Tsukamoto, N.: An Empirical Study on Similarity-Based Mating for Evolutionary Multiobjective Combinatorial Optimization. Eur. J. Oper. Res. 188(1), 57–75 (2008)
3. Kim, Y., Kim, J.H., Han, K.H.: Quantum-Inspired Multiobjective Evolutionary Algorithm for Multiobjective 0/1 Knapsack Problems. In: Proc. 2006 IEEE Congress on Evolutionary Computation, Vancouver, Canada, pp. 2601–2606. IEEE Press, Los Alamitos (2006)
4. Li, B.B., Wang, L.: A Hybrid Quantum-Inspired Genetic Algorithm for Multiobjective Flow Shop Scheduling. IEEE Trans. Evol. Comput. 37(3), 576–591 (2007)
5. Gao, J.Q., Wang, J.: WBMOAIS: A Novel Artificial Immune System for Multiobjective Optimization. Comput. Oper. Res. 37(1), 50–61 (2010)
6. Gao, J.Q., Fang, L., Wang, J.: A Weight-Based Multiobjective Immune Algorithm: WBMOIA. Eng. Optimiz. (2009), doi:10.1080/03052150903406563
7. Han, K.H., Kim, J.H.: Quantum-Inspired Evolutionary Algorithms With a New Termination Criterion, H_ε Gate, and Two-Phase Scheme. IEEE Trans. Evol. Compt. 8(2), 156–169 (2004)
8. Li, H., Zhang, Q.F., Tsang, D., Ford, J.A.: Hybrid Estimation of Distribution Algorithm for Multiobjective Knapsack Problem. In: Gottlieb, J., Raidl, G.R. (eds.) EvoCOP 2004. LNCS, vol. 3004, pp. 145–154. Springer, Heidelberg (2004)

RBF Neural Network Based on Particle Swarm Optimization

Yuxiang Shao[1], Qing Chen[2,3], and Hong Jiang[2,3]

[1] School of Computer Sciences, China University of Geosciences,
Wuhan, Hubei, 430074, China
syxcq@163.com
[2] Hubei Province Key Laboratory of Intelligent Robot, Wuhan Institute of Technology,
Wuhan, Hubei, 430073, China
[3] School of Computer Science & Engineering, Wuhan Institute of Technology,
Wuhan, Hubei, 430073, China

Abstract. This paper develops a RBF neural network based on particle swarm optimization (PSO) algorithm. It is composed of a RBF neural network, whose parameters including clustering centers, variances of Radial Basis Function and weights are optimized by PSO algorithm. Therefore it has not only simplified the structure of RBF neural network, but also enhanced training speed and mapping accurate. The performance and effectiveness of the proposed method are evaluated by using function simulation and compared with RBF neural network. The result shows that the optimized RBF neural network has significant advantages inspect of fast convergence speed, good generalization ability and not easy to yield minimal local results.

Keywords: RBF neural network, Particle swarm optimization algorithm, Optimize.

1 Introduction

Radial basis function (RBF), emerged as a variant of artificial neural network, have been successfully applied to a large diversity of applications including interpolation, chaotic time-series modeling, control engineering, image restoration, data fusion, etc[1~3].

In RBF network, parameters of basis functions (such as width, the position and number of centers) in the nonlinear hidden layer have great influence on the performance of the network[4][5]. Common RBF training algorithms cannot possibly find the global optima of nonlinear parameters in the hidden layer, and often have too many hidden units to reach certain approximation abilities, which will lead to too large a scale for the network and decline of generalization ability.

Particle Swarm Optimization (PSO) algorithm is a global optimization technology based on the group intelligence, it carries on the intelligent search for the solution space through mutual effect in order to discover the optimal solution[6][7].

In this paper, a hybrid RBF training method combining Particle Swarm Optimization (PSO) algorithm is proposed. In this method, PSO algorithm is used to determine

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 169–176, 2010.

the structure and parameters in RBF hidden layer, and RBF linear output weights as well. The experiments showed that this method improved effectively the convergence speed of the algorithm and overcomes the problem of immature convergence, and the proposed method integrating RBF network and PSO algorithm are effective and feasible.

This paper is divided into four sections. In Sect. 2, the integration of RBF network and PSO algorithm is described. In Sect. 3, simulation experiment through two functions is done and the results are presented. The conclusions are given in Sect. 4.

2 Integration of RBF Network and PSO Algorithm

2.1 Radial Basis Function Neural Network

Radial basis function (RBF) neural network was proposed by Broomhead and Lowe, and this neural network differs from neural networks with sigmoidal activation functions in that it utilizes basic functions in the hidden layer that are locally responsive to input stimulus. RBF are embedded in a two-layer neural network, where each hidden unit implements a radial activated function. The output units implement a weighted sum of hidden unit outputs. While the input into a RBF network is nonlinear, the output is often linear. Their excellent approximation capabilities have been studied by Park and Sandberg. Owing to their nonlinear approximation properties, RBF networks are able to model complex mappings, indicating that neural networks can only model by means of multiple intermediary layers.

2.2 Radial Basis Function Network Model

The RBF network topological structure is shown in Fig.1. The network consists of three layers, namely the input layer, radial basic function hidden layer and output layer. The input part does not transform the signals but only dispatches the input vector to the radial basic layer. The function in a hidden layer node (also called nucleus function) responds partly to the input signals, i.e. when the input function is close to the center range of the nucleus function, the hidden layer will produce a larger output. The output layer makes output values through a linear combination of outputs from the hidden layer.

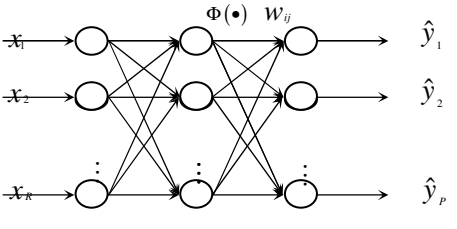


Fig. 1. Structure of RBF neural network

Here input vector $X=[x_1,x_2,...,x_R]$; c_i - the center of RBF neural network, a constant vector with the same dimension as X; R-the dimension of the input vector; M-neurons number of the hidden layer; $\Phi(\cdot)$ -radial basis function; $\|X-c_i\|$ - Euclidean distance between X and c_i; j-output node, j=1,2 ..... P; w_{ij} -the weight value which connected the i-th hidden node with the j-th output node.

As shown in figure l, ideal output y_j (j=1,2, . . . ,P), the actual output $\hat{y}_j$ and the weight value of the output layer w_{ij} can be obtained by the RBF neural network. Choosing Gauss function $\Phi_i(x)=\exp\left(-\frac{\|x-c_i\|^2}{2\sigma_i^2}\right)$ as radial basis function, the actual output $\hat{y}_j$ is calculated by the following formula:

$$\hat{y}_j=\sum_{i=1}^{M}w_{ij}\Phi_i(x)=\sum_{i=1}^{M}w_{ij}\exp\left(-\frac{\|X-c_i\|^2}{2\sigma_i^2}\right) \tag{1}$$

Then, the weight value w_{ij} is adjusted to satisfy the following formula, from which the final result of the RBF neural network can be obtained.

$$E=\sum_{j=1}^{P}\left(y_j-\hat{y}_j\right)^2=\sum_{j=1}^{P}\left(y_j-\sum_{i=1}^{M}w_{ij}\Phi_i(x)\right)^2 \tag{2}$$

2.3 PSO Algorithm

PS0 algorithm is initialized with a group of random solutions(called swarms). Each potential solutions, called particles, fly through the problem space at velocities following those of the current optimum particles and searches for optima by updating generations [8][9]. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) achieved so far (fitness value is also stored). This value is called P_i. Another "best" value tracked by the particle swarm optimizer is the best value obtained so far by any particle in the population. When this best value is a global best, it is called P_g. The PSO concept consists of, at each time step, changing the velocity (accelerating) of each particle toward its P_i and P_g locations (global version of PSO). Finally the results are achieved [10].

The particle i updates its velocity and location according to the following formulae:

$$V_i^{k+1} = wV_i^k + c_1 r_1 \left(p_i - X_i^k \right) + c_2 r_2 \left(p_g - X_i^k \right) \tag{3}$$

$$X_i^{k+1} = X_i^k + \lambda V_i^{k+1} \tag{4}$$

where X_i is the position of the particle i; V_i represents the velocity of the particle i, $V_i \in (v_{\min}, v_{\max})$; $v_{\min}$ and $v_{\max}$ are constants in the range of [0,1]; P_i is the best previous position of particle i; $w \in R$ is called inertia weight and $\lambda \in R$ is called constriction factor which is used alternatively to $w \in R$ to limit velocity; $c_1 \in R$ and $c_2 \in R$ are acceleration constants; r_1 and r_2 are random real numbers in the range of [0,1]. When the given maximum iteration frequency or fitness value meets the requirement to realize the given value, the calculation is terminated.

The inertia weight is set to the following equation :

$$w = w_{\max} - \left((w_{\max} - w_{\min}) / iter_{\max} \right) \times iter \tag{5}$$

where $w_{\max}$ is the initial value of weighting coefficient; $w_{\min}$ is the final value of weighting coefficient; $iter_{\max}$ is the maximum number of iterations or generation; $iter$ is the current iteration or generation number.

2.4 Algorithm Flow

In this paper, PSO algorithm uses real-coded, makes c_i, σ_i and w_{ij} of RBF network as a particle. The whole integration step is summarized as following:

Step 1: Initialize swarm, including swarm size, each particle's position and velocity; Give initial value: $w_{\max}$, $w_{\min}$, and generation=0.

Step 2: Set the range of w_{ij} is $\left[w'_{\min}, w'_{\max} \right]$. makes c_i, σ_i and w_{ij} of RBF network as a particle. Thus, build the initial swarm.

Step 3: Calculate individual fitness, decode the individuals, assign them to c_i, σ_i and w_{ij} of RBF network. Calculate the study samples' output error. The fitness of particle a is defined as:

$$f(a) = \sum_{i=1}^{P} \sum_{j=1}^{n} \left(y_{ij} - t_{ij} \right)^2 \tag{6}$$

where y_{ij} is the calculated output of individual network i, t_{ij} is the expected output, n is the number of training set examples, and P is the number of output nodes.

Step 4: Determine whether meet the conditions to terminate the algorithm.

Step 5: If it meets the condition, it will go to step 6, else generate next swarm by PSO algorithm, find new P_i and P_g, update P_g of the swarm and P_i of each particle, go to step 3.

Step 6: Decode the optimal individual searched by PSO algorithm, assign them to c_i, σ_i and w_{ij} of RBF network.

3 Simulation Experiment

In this section, we will use two different examples to validate the new method. We do experiments with two different algorithms, RBF network and RBF network optimized by PSO algorithm. For the two algorithms, we adopt same initialization conditions.

Example 1: $y = \sin(5x) + \cos(3x),\ x \in [-1,1]$

From the concrete simulation we obtain Figure 2 to Figure 4.

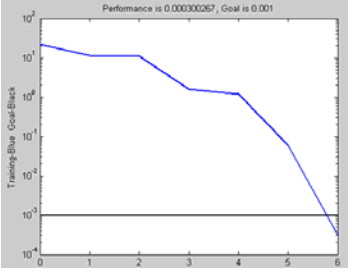


Fig. 2. Training curve of RBF network optimized by PSO algorithm

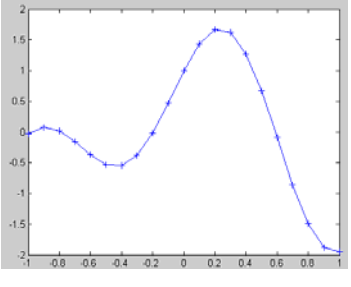


Fig. 3. Simulation result of RBF network optimized by PSO algorithm

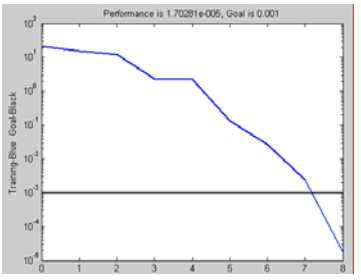


Fig. 4. Training curve of RBF network

Example 2: $f(x,y)=(1-x)^2+100(y-x^2)^2, x\in[0,1], y\in[0,1]$

From the concrete simulation we obtain Figure 5 to Figure 7.

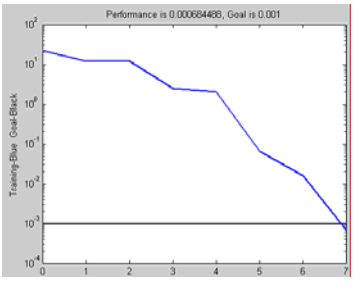


Fig. 5. Training curve of RBF network optimized by PSO algorithm

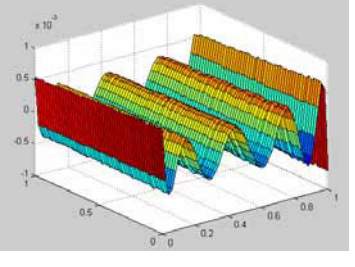


Fig. 6. Error surface of RBF network optimized by PSO algorithm

The two examples are utilized to validate the proposed new method. From all above figures, we can learn that RBF neural network system which is optimized by PSO algorithm has better convergence rate and higher learning precision. Meanwhile RBF neural network system which is optimized by PSO algorithm can obtain better simulation results compared with RBF neural network. The simulation shows the feasibility and validity of the new method.

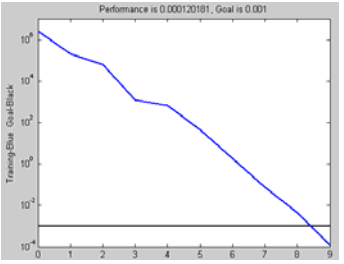


Fig. 7. Training curve of RBF network

4 Conclusions

In this paper, PSO algorithm is introduced to improve the behavior of RBF network. In view of the inherent defects in the tradition learning algorithm of RBF NN, PSO algorithm optimizes c_i, σ_i and w_{ij} of RBF network. Then the proposed method is applied to function simulation and compared with RBF network. The numerical results show that the RBF neural network system which is optimized by PSO algorithm has a better training performance, faster convergence rate than the RBF network. It is worth to mention that the current study is very preliminary for the RBF neural networks approach, and there are many works need to be carried on further.

Acknowledgement. The Project was Supported by the Special Fund for Basic Scientific Research of Central Colleges, China University of Geosciences (Wuhan) (No. CUGL090237) and the 863 Program of China (No. 2008AA121602).

References

1. Watanabe, T.: Transformation of Spectral Envelope for Voice Conversion Based on Radial Basis function Networks. In: 8th International Conference on Spoken Language Processing, pp. 285–288. IEEE Press, Denver (2002)
2. Chen, S., Cowan, C.F.N., Grant, P.M.: Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks. J. IEEE Transactions on Neural Networks 3, 302–308 (1991)
3. Yu, C.-g., Ying, Y.-b., Wang, J.-p.: Determining Heating Pipe Temperature in Greenhouse Using Proportional Integral Plus Feedforward Control and Radial Basic Function Neural Networks. J. Journal of Zhejiang University SCIENCE 6A(4), 265–269 (2005)
4. Chen, S., Grant, P.M., Cowan, C.F.N.: Orthogonal Least-Squares Algorithm for Training Multioutput Radial Basis Function Networks. IEEE Transactions on Neural Networks 6, 378–384 (1992)
5. Knopf, G.K., Sangole, A.: Interpolating Scattered Data Using 2D Self-Organizing Feature Maps. J. Graphical Models 66, 50–69 (2004)
6. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neutral Networks, pp. 1942–1948. IEEE Press, Perth (1995)

7. Shi, Y.H., Eberhart, R.C.: Modified Particle Swarm Optimizer. In: IEEE International Conference on Evolutionary computation, pp. 69–73. IEEE Press, Anchorage (1998)
8. Eberhart, R.C., Shi, Y.H.: Particle Swarm Optimization and Its Applications to VLSI Design and Video Technology. In: Proceedings of the 2005 IEEE International Workshop on VLSI Design and Video Technology, pp. 29–39. IEEE Press, Suzhou (2005)
9. Shi, Y.H., Eberhart, R.C.: A Hybrid Self-Organizing Maps and Particle Swarm Optimization Approach. Concurrency Computation Practice and Experience 16, 895–915 (2004)
10. Zhao, J.Y., Jia, L., Yang, L.C.: RBF Neural Network Traffic Flow Forecasting Model Based on Particle Swarm Optimization. Journal of Highway and Transportation Research and Development 23(7), 116–119 (2006)

Genetic-Based Granular Radial Basis Function Neural Network

Ho-Sung Park[1], Sung-Kwun Oh[2], and Hyun-Ki Kim[2]

[1] Industry Administration Institute, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
parkhs@suwon.ac.kr
[2] Department of Electrical Engineering, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
{ohsk,hkkim}@suwon.ac.kr
http://autosys.suwon.ac.kr

Abstract. In this paper, we introduce a new architecture of GA-based Granular Radial Basis Function Neural Networks (GRBFNN) and discuss its comprehensive design methodology. The architecture of the network is fully reflective of the structure encountered in the training data which are granulated with the aid of clustering techniques. More specifically, the output space is granulated with use of K-means clustering while the information granules in the multidimensional input space are formed by using a so-called context-based Fuzzy C-means which takes into account the structure being already formed in the output space. The GA-based design procedure being applied to each receptive fields of GRBFNN leads to the selection of preferred receptive fields with specific local characteristics (such as the number of context, the number of clusters for each context, and the input variables for each context) available within the GRBFNN.

Keywords: Granular radial basis function neural networks (GRBFNN), Context-based fuzzy C-means, Genetic algorithm, Machine learning data.

1 Introduction

The current trend in intelligent systems design and analysis or Computational Intelligence research is concerned with the integration of its underlying tools (say, fuzzy technology [1], neural networks [2], evolutionary algorithm [3], etc.) and data processing tools (K-means clustering, fuzzy C-means clustering [4], etc.) in a hybrid framework for solving complex problems. In particular, radial basis function neural networks (RBFNN) have been widely studied and applied to many categories of problems such as those arising in pattern recognition, signal processing, time series prediction, and nonlinear system modeling and system control, cf. [5].

In this research, our main objective is to develop a design strategy of GA-based Granular Radial Basis Function Neural Network (GRBFNN) in which (a) an architecture of the network is fully reflective of the structure encountered in the training data being granulated with the aid of fuzzy clustering, and (b) an optimal parameters design available within receptive fields (viz. the number of context, the number of

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 177–184, 2010.

clusters for each context, and the input variables for each context) lead to a structurally and parametrically optimized network by the genetic algorithm.

2 General RBF Neural Network

RBF neural network [5], [6] is a three-layer neural network, in which an n-dimensional input vector $\mathbf{x}=[x_1, x_2, ..., x_n]^T$ is transformed in a nonlinear fashion by a series of the receptive fields. Quite commonly, the receptive fields are described by the Gaussian basis functions of the form

$$\Phi_i(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{v}_i\|^2}{2r_i^2}\right) \tag{1}$$

The output of the network $y(\mathbf{x})$ is computed as a linear combination of the individual activation levels, that is

$$y(\mathbf{x}) = \sum_{i=1}^{K} w_i \Phi_i(\mathbf{x}) \tag{2}$$

3 The Data Processing by Granular Computing

3.1 K-Means Clustering in the Output Space

K-means clustering has been intensively used not only to organize and categorize data, but it becomes useful in data compression and model identification [7].

In this setting, the K-means algorithm partitions a collection of output data, that is $\{target_k\}$, k=1,2, .., N into P-2 clusters and determines prototypes of each cluster such that a certain cost function (performance index) V is minimized.

$$V = \sum_{i=2}^{P-1} \sum_{k=1}^{N} (target_k - y_i)^2 \tag{3}$$

We also treat the minimal and maximal value in these output data as the two additional prototypes. This in total gives rise to P prototypes. Denote these prototypes by y_1, y_2, …, and y_P, respectively. Over these prototypes we span triangular membership functions with the modal values positioned at y_i's. There is an ½ overlap between two successive fuzzy sets. We denote the membership functions obtained in this manner by T_1, T_2, …, and T_P, respectively. In particular, the membership degree of the data $\{target_k\}$ in the j-th context fuzzy set is denoted by t_{jk}.

3.2 Context-Based Fuzzy C-Means Clustering in the Input Space

The context-based clustering supporting the design of information granules is completed in the space of the input data while the build of the clusters is guided by a collection of some predefined fuzzy sets (so-called contexts) defined in the output space, cf. [6], [8]. Let us introduce a family of the partition matrices induced by the j-th context and denote it by $\mathbf{U}(T_j)$,

$$\mathbf{U}(T_j) = \left\{ u_{ik} \in [0,1], \quad \sum_{i=1}^{c} u_{ik} = t_{jk} \ \forall k, \quad 0 < \sum_{k=1}^{N} u_{ik} < N \ \forall i \right\} \tag{4}$$

The objective function of the context-based clustering is defined as follows:

$$V = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}^{m} \left\| \mathbf{x}_k - \mathbf{z}_i \right\|^2 \tag{5}$$

The minimization of V is realized under the constraints expressed by (5) so in essence we end up with "P" separate clustering tasks implied by the respective contexts. Briefly speaking we have

$$\text{Min V subject to } \mathbf{U}(T_j), \quad j = 1, 2, \cdots, P \tag{6}$$

The minimization of V as completed by the context-based FCM is realized by iteratively updating the values of the partition matrix and the prototypes.

The update of the partition matrix and the prototypes ($\mathbf{z}_1$, $\mathbf{z}_2$, …, $\mathbf{z}_c$) are completed as follows

$$u_{ik} = \frac{t_{jk}}{\sum_{l=1}^{c} \left(\frac{\left\| \mathbf{x}_k - \mathbf{z}_i \right\|}{\left\| \mathbf{x}_k - \mathbf{z}_l \right\|} \right)^{2/(m-1)}}, \quad \mathbf{z}_i = \frac{\sum_{k=1}^{N} (u_{ik})^m \cdot \mathbf{x}_k}{\sum_{k=1}^{N} (u_{ik})^m}, \ i=1, \ldots c, \ k=1, \ldots, N. \tag{7}$$

4 The Architecture of the Granular RBF Neural Network

The network dwells on the concept of context-based clustering method. The fuzzy partitions formed for all variables gives rise to the topology as visualized Figure 1.

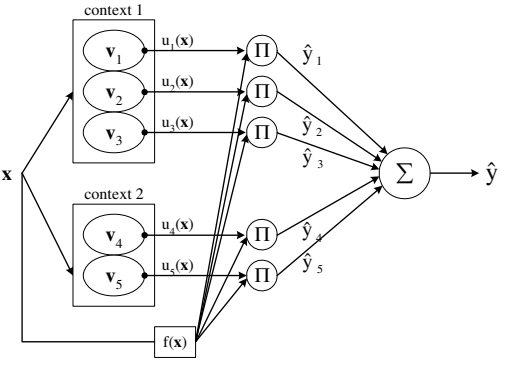


Fig. 1. Structure of the proposed RBFNN

Alluding to the terminology of fuzzy rule-based systems, this structure translates into the following collection of rules:

$$R^i : \text{If } \mathbf{x} \text{ is } "p\text{-th context}" \text{ and } "c\text{-th cluster}" \quad \text{then } \hat{y}_i = y_p + \mathbf{a}_i^T(\mathbf{x}^p - \mathbf{v}_c) \tag{8}$$

where, $\hat{y}_i$ is i-th local model (i=1, 2, …, P×c), y_p is the modal value of the p-th context, $\mathbf{x}^p$ is the selected input set of p-th context. R^i stands for the i-th rule.

When taking all rules into account, we arrive at the expression governing the output of the network

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{c_1} u_i(\mathbf{x})\left[y_1 + \mathbf{a}_i^T(\mathbf{x}^1 - \mathbf{v}_i)\right] + \sum_{i=c_1}^{c_2} u_i(\mathbf{x})\left[y_2 + \mathbf{a}_i^T(\mathbf{x}^2 - \mathbf{v}_i)\right] + \ldots + \sum_{i=(c_1+c_2+\cdots c_{p-1})+1}^{c_p} u_i(\mathbf{x})\left[y_P + \mathbf{a}_i^T(\mathbf{x}^p - \mathbf{v}_i)\right] \tag{9}$$

The performance index used in the evaluation of the network comes in the form of the Root Mean Square Error (RMSE). The standard Lest Square Error (LSE) is used to determine the coefficients of each of the local models.

5 Optimization by Genetic Algorithms

Genetic algorithms support robust and population-oriented search realized in complex search spaces [3]. Proceeding with the optimization details, the GA use the serial method of float type, roulette-wheel in the selection operator, one-point crossover in the crossover operator, and invert operator in the realization of the mutation operator. To reduce the stochastic diversity of roulette-wheel selection, we use elitist strategy. The fitness function that is an essential component of any environment of genetic computing is chosen so that reflects the performance of the network. In what follows, we consider the following form of the fitness function to be maximized:

$$\text{Fitness function} = 1/\text{PI} \tag{10}$$

6 Experimental Results

The proposed architecture of the network, its development and resulting performance are illustrated with the aid of a series of numeric experiments. In all experiments, we randomly divide the data set into the training (60%) and testing (40%) part of data, respectively. Table 1 includes a list of parameters used in the genetic optimization of the network used in the numeric experiments.

Table 1. Parameters used in the genetic optimization of the nework

	Synthetic 3-D data	Automobile MPG data
Generation size	100	
Population size	100	
Crossover size	0.65	
Mutation rate	0.1	
Number of all input variables	3	7

6.1 Synthetic Three-Dimensional Data

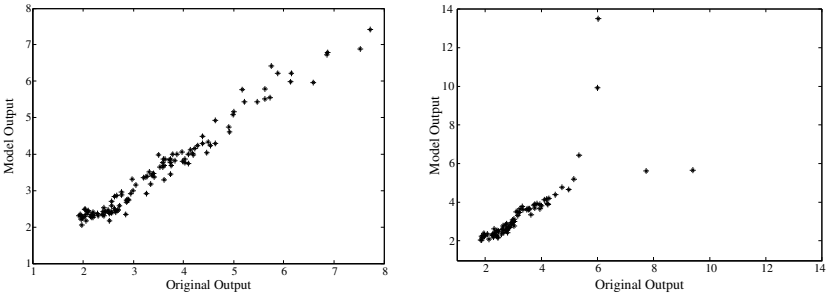
We consider a nonlinear three-argument (variable) function of the following form

$$y = f(x_1, x_2, x_3) = \left(1 + x_1^{-2} + x_2^{-1.5} + x_3^{-1.0}\right)^2, \quad 1 \le x_1, x_2, x_3 \le 5 \tag{11}$$

Using (11), we generated 200 input-output data pairs. The experiments were carried out for the number of contexts varying in-between 2 and 6. Table 2 summarizes the lowest values of the performance index obtained through the proposed methodology.

Table 2. Values of the performance index for the synthetic three-dimensional data

No. of selected input variables per context	Selected number of clusters per context						Selected input variables for each context						PI	EPI
	1th	2nd	3rd	4th	5th	6th	1th	2nd	3rd	4th	5th	6th		
1	2	2	3	4	3	5	2	2	1	2	3	2	0.1321	0.3051
2	5	4	2	5	5		1 3	1 3	1 2	2 3	1 2		0.1208	0.1851
3	3	4	3	3	3	4	1, 2, 3						0.0544	0.3116



(a) training data (b) testing data

Fig. 2. Scatter plots of data versus the output of the network

The best performance (quantified as PI=0.1208 and EPI=0.1851) is reported for the network with 5 contexts and {5, 4, 2, 5, 5} clusters per each context. In the case all input variables were used, the performance of the best model comes with PI being equal to 0.0544 and EPI given as 0.3116. Figure 2 visualizes the approximation and generalization capabilities of the network (here PI=0.1028, EPI=0.1851).

Figure 3 shows the performance index (RMSE) of the optimized network obtained for the number of selected input variables per context.

Table 3 summarizes the performance of the proposed network vis-à-vis the results produced by the linear regression model and the "standard" RBF neural network.

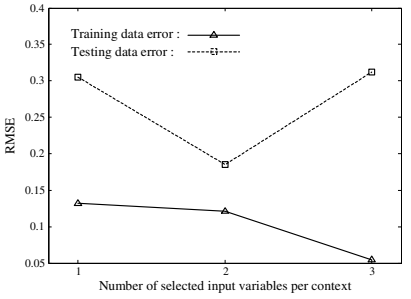


Fig. 3. Performance index regarded as a number of selected input variables per context

Table 3. Comparative analysis of the performance of selected model

Model	Number of selected input variables	Performance index	
		PI	EPI
Linear regression		0.6385±0.0374	0.7447±0.0513
RBFNN [9]	Number of nodes =5	0.8732±0.0869	0.9964±0.1316
	Number of nodes=15	0.4693±0.0547	0.7695±0.1148
	Number of nodes=30	0.2731±0.0210	0.5663±0.1117
Proposed model	1	0.1321	0.3051
	2	0.1208	0.1851
	3	0.0544	0.3116

6.2 Machine Learning Data

We consider the well-known automobile MPG data with the output being the automobile's fuel consumption expressed in miles per gallon. Here we consider 392 input-output pairs. The best results are reported in Table 4.

Table 4. Performance index of the proposed model

No. of selected input per context	Selected number of clusters per context						Selected input variables for each context						PI	EPI
	1th	2nd	3rd	4th	5th	6th	1th	2nd	3rd	4th	5th	6th		
1	4	4	3	5	5		5	4	3	6	4		1.1100	1.8151
2	4	4	2	4	4	2	3 5	3 5	1 4	4 6	5 7	3 6	0.7962	1.1704
3	5	4	4	2	4	4	4 5 7	1 2 7	1 2 3	3 6 7	1 3 4	3 4 6	0.6659	1.0976
4	4	2	4	3	4	3	1 3 4 7	2 4 5 6	1 2 5 7	1 2 6 7	2 5 6 7	1 2 3 7	0.6065	1.2789
5	5	4	2	2	4	2	3 4 5 6 7	1 2 4 6 7	1 2 3 5 7	1 2 5 6 7	2 3 4 6 7	3 4 5 6 7	0.6673	1.4784
6	3	3	3	3	3	3	1 2 3 4 5 7	1 2 3 4 5 6	1 3 4 5 6 7	1 3 4 5 6 7	1 2 4 5 6 7	1 2 4 5 6 7	0.9486	1.3026

Figure 4 presents the values of the performance index.

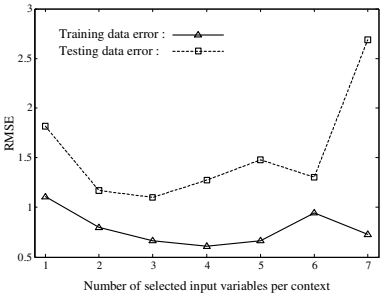


Fig. 4. Performance index as increase of a number of selected input variables per contexts

Table 5 summarizes the performance of the proposed network versus other models.

Table 5. Comparative analysis of the performance of selected model

Model	Number of selected input variables	Performance index	
		PI	EPI
Linear regression		3.1609±0.0716	3.5701±0.1032
RBFNN [9]	Number of nodes =5	7.2746±0.1457	7.7341±0.2382
	Number of nodes =15	6.6949±0.1711	7.5651±0.3660
	Number of nodes =30	5.7557±0.0722	7.7140±0.1920
Proposed model	1	1.1100	1.8151
	2	0.7962	1.1704
	3	0.6659	1.0976
	4	0.6065	1.2789
	5	0.6673	1.4784
	6	0.9486	1.3026
	7	0.7306	2.6903

7 Concluding Remarks

In this paper, the GA-based design procedure of Granular Radial Basis Function Neural Network (GRBFNN) and its design methodology were proposed to construct optimal model architecture for nonlinear and complex system modeling. The information granules are developed using a certain context-driven version of the Fuzzy C-means. This specialized clustering environment emphasizes the role of contexts – fuzzy sets defined in the output space in the formation of the information granules in the input space. And GA-based design procedure at each context of GRBFNN leads to the selection of these preferred parameters available within receptive field, and then based on these selections, we build the flexible and optimized architecture of the GA-based GRRBNN.

Acknowledgements. This work was supported by the Korea Research Foundation Grant funded by the Korean Government [KRF-2008-359-D00007] and also supported by the GRRC program of Gyeonggi province [GRRC SUWON2009-B2, Center for U-city Security & Surveillance Technology].

References

1. Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and process. IEEE Trans. Syst. Man Cybern. 1, 28–44 (1973)
2. Jang, J.S.R.: ANFIS: Adaptive-Network-Based Fuzzy Inference Systems. IEEE Trans. Syst. Man Cybern. 23, 665–685 (1993)
3. Goldberg, D.E.: Genetic Algorithm in Search, Optimization & Machine. Addison-Wesley, Reading (1989)
4. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithm. Plenum, New York (1981)
5. Pedrycz, W.: Conditional Fuzzy Clustering in The Design of Radial Basis Function Neural Networks. IEEE Trans. Neural Network. 9, 601–612 (1998)
6. Pedrycz, W., Park, H.S., Oh, S.K.: A granular-oriented development of functional radial basis function neural networks. Neurocomputing 72, 420–435 (2008)
7. Park, H.S., Pedrycz, W., Oh, S.K.: Evolutionary design of hybrid self-organizing fuzzy polynomial neural networks with the aid of information granulation. Expert Systems with Applications 33, 830–846 (2007)
8. Pedrycz, W.: Conditional Fuzzy C-Means. Pattern Recogn. Letter. 17, 625–631 (1996)
9. http://www.mathworks.com/access/helpdesk/help/toolbox/nnet

A Closed-Form Solution to the Problem of Averaging over the Lie Group of Special Orthogonal Matrices

Simone Fiori

Dipartimento di Ingegneria Biomedica, Elettronica e Telecomunicazioni (DiBET)
Università Politecnica delle Marche, Via Brecce Bianche, Ancona I-60131, Italy
s.fiori@univpm.it
http://web.dibet.univpm.it/fiori

Abstract. Averaging over the Lie group $SO(p)$ of special orthogonal matrices has several applications in the neural network field. The problem of averaging over the group $SO(3)$ has been studied in details and, in some specific cases, it admits a closed form solution. Averaging over a generic-dimensional group $SO(p)$ has also been studied recently, although the common formulation in terms of Riemannian mean leads to a matrix-type non-linear problem to solve, which, in general, may be tackled via iterative algorithms only. In the present paper, we propose a novel formulation of the problem that gives rise to a closed form solution for the average $SO(p)$-matrix.

Keywords: Averaging over curved manifolds; Closed-form solution; Learning theory; Special Orthogonal matrices.

1 Introduction

Averaging is a rather common way to smooth out data and get rid of noise or measurement errors. Computing the mean value or the median value of a set of scalar, real-valued measures is an easy task. In recent engineering applications, however, data to process do not appear as simple real scalars, but they may appear as structured data, like vectors or matrices, whose entries should further satisfy nonlinear constraints.

An interesting case of averaging is the one in which the data are multidimensional rotations, namely, they belong to the Lie group of special orthogonal matrices $SO(p)$. Averaging and interpolation over the group $SO(p)$ has several applications.

An early application was to the study of plate tectonics [8]. In geophysics, estimates of the relative motion of continental plates among various geological epochs are available. For a given pair of plates, such a relative movement may be summarized by a rotation matrix that describes the orientation of a plate with respect to its initial position. The general problem of rotation estimation at non-observation times may be addressed by interpolation in the group of rotations.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 185–192, 2010.

An interesting application is to invariant visual perception [9]. One of the most important problems in visual perception is the one of visual invariance, namely, how objects are perceived to be the same despite undergoing transformations such as translations, rotations or scaling. It was shown that approaches based on first-order Taylor series expansions of inputs may be regarded as special cases of the $SO(p)$-based approach, the latter being capable of handling arbitrarily large transformations in contrast to those based on truncated series expansions.

Another noticeable application was found in the modeling of DNA chains [6]. Since the discovery of the double helical structure of DNA chains, this molecule has been the focus of study by numerous scientists. Mechanical models of DNA have offered a new perspective for studying its macroscopic properties. In particular, elastic-rod models have been widely used as an approximation to DNA actual mechanical structure. The elastic-rod model consists of a framed curve, or centerline of the rod, that runs through the middle of the double helix. Modeling of DNA sequences may be effected thus by three-dimensional rods, whose configuration may be described by $SO(3)$-valued functions. In fact, the configuration of a rod may be described by a centerline and a function of directors that form an orthogonal frame giving the orientation of the cross-section of the rod at a given location. In the context of DNA modeling, experimental observations are obtained with a significant amount of noise that needs to be smoothed out, e.g., by averaging.

A well-known application is data compression/denoising by principal component analysis [3]. Data compression may be achieved by seeking for a change of basis that concentrate the energy of the data in a few directions and leave a little fraction of the data energy in the complementary directions. In the case of data denoising, it is the noise components that make a low-dimensional data-set appear as a high-dimensional data-set and that may be partially discarded by projecting the data over a suitable basis. Such basis is orthogonal and may be described by a special orthogonal matrix.

Another well-known application is to blind source separation and independent component analysis [4]. Blind separation of signals in telecommunication, acoustics and medical imagery analysis may be obtained, upon normalization of data covariance matrix, by the help of a high-dimensional rotation that may be represented by a special orthogonal matrix to be learnt.

The case of averaging over the group $SO(3)$ has been studied in details [7] and it is known that in some specific cases it admits a closed form solution. The more general problem of averaging over a higher-dimensional group $SO(p)$, with $p > 3$, has also been studied recently [5], although the common formulation in terms of Riemannian mean leads to a matrix-type non-linear problem to solve, which may be tackled via iterative algorithms only (gradient/based or Newton-like) (see, e.g., [2]).

In the present paper, we propose a novel formulation of the problem that gives rise to a closed form solution for the average special orthogonal matrix. The key point is to endow the Lie group $SO(p)$ with an alternative group structure with respect to the one arising by matrix multiplication/inverse.

2 General Setting and Averaging Algorithm Design

In the following subsection, we briefly recall some properties of Lie groups and consider the problem of averaging over a Lie group in general. For a reference of differential geometry see, e.g., [10].

2.1 Algebraic and Geometric Setting

A Lie group is an algebraic group that also possesses the structure of a smooth manifold, namely, which is locally diffeomorphic to an Euclidean space.

An algebraic group structure (G, m, i, e) is made of a set G endowed with a multiplication operation m, an inverse operation i and an identity element e, such that for every $g_1, g_2 \in G$, it holds $m(g_1, g_2) \in G$, $m(g_1, i(g_1)) = m(i(g_1), g_1) = e$ and $m(g_1, e) = m(e, g_1) = g_1$. The group identity and inverse need to be unique and the group multiplication needs to be associative, namely, it must hold $m(g_1, m(g_2, g_3)) = m(m(g_1, g_2), g_3)$, for every $g_1, g_2, g_3 \in G$.

In the theory of Lie groups, the group G is endowed with a differential manifold structure. The tangent space of the manifold G at a point $g \in G$ is denoted by T_gG. The tangent space at the identity of a Lie group plays a prominent role in the theory of Lie groups. It is termed Lie algebra as is denoted by $\mathfrak{g} \stackrel{\text{def}}{=} T_eG$.

Further to the above properties, it is necessary to ensure that the algebraic and differential structures be compatible, namely, that the application $(g_1, g_2) \mapsto m(g_1, i(g_2))$ be infinitely differentiable for every $g_1, g_2 \in G$.

A left translation $\ell : G \times G \to G$ may be associated to the Lie group G, which is defined by $\ell_g(g_1) = m(i(g), g_1)$ for all $g, g_1 \in G$. The inverse of operator $\ell_g(\cdot) : G \to G$ is $\ell_g^{-1}(g_1) = m(g, g_1)$.

Exponential maps may be associated to a manifold. The exponential map $\exp_g : T_gG \to G$ pulls down a tangent vector to the manifold and its inverse operator is denoted by $\log_g$, which maps a point of G into a point in T_gG. Exponential and logarithmic maps at the identity of the Lie group G, namely maps $\exp_e$ and $\log_e$, are simply denoted as exp and log, respectively.

2.2 Averaging over a Lie Group

In a vector space $V \subset \mathbb{R}^p$, averages may be calculated in the arithmetic sense, namely, if $v_1, \ldots, v_N \in V$, then a possible average formula to compute a mean value $v \in V$ is:

$$v \stackrel{\text{def}}{=} \frac{1}{N}(v_1 + v_2 + \cdots + v_N) \ . \tag{1}$$

This is possible because in vector spaces that are subsets of $\mathbb{R}^p$, multiplication by scalars and addition are possible. In a Lie group, however, the term 'mean value' does not carry on the usual weighted-sum meaning because – in general – a matrix Lie group G is a curved manifold, so if $g_1, g_2 \in G$ then $g_1 + g_2 \notin G$, where symbol + stands for matrix-to-matrix addition.

In order to compute averages on a Lie group, it is possible to take advantage of the exponential/logarithmic maps to push-up every data point from the group G to its Lie algebra $\mathfrak{g}$, which is a *vector space* where averaging can be performed in the arithmetic sense, and then to shift back the mean Lie-algebra vector to the group. The following logical succession of steps leads to an equation characterizing the mean Lie-group matrix $g \in G$ of a set of points $g_1, \ldots, g_N \in G$ [5]:

1. The first step is to shift the points $g_n \in G$ in a neighborhood of the identity $e \in G$ by the help of a left-translation about the (unknown) mean matrix g. The shifted points are given by $\ell_g(g_n) = m(i(g), g_n) \in G$, $n \in \{1, \ldots, N\}$.
2. As the points $\ell_g(g_k)$ belong to a neighborhood of the identity of the group G, they may be pushed-up to the Lie algebra $\mathfrak{g}$ by applying the logarithmic map operator, which gives rise to points $L_n \stackrel{\text{def}}{=} \log(m(i(g), g_n)) \in \mathfrak{g}$, $n \in \{1, \ldots, N\}$.
3. In the Lie algebra $\mathfrak{g}$, averaging may be computed in the arithmetic sense, namely by $L \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^{N} L_n$.
4. The arithmetic mean $L \in \mathfrak{g}$ must correspond, up to pulling-down to the Lie group and inverse left-translation, to the average point $g \in G$, so it must hold $\ell_g^{-1}(\exp(L)) = g$.

In summary, the equation that characterizes the mean Lie-group matrix $g \in G$ may be written as:

$$g = m\left(g, \exp\left(\frac{1}{N} \sum_{n=1}^{N} \log(m(i(g), g_n))\right)\right) \ . \tag{2}$$

Apparently from equation (2), the fixed point g is characterized by the fact that $\exp(\frac{1}{N} \sum_{n=1}^{N} \log(m(i(g), g_n))) = e$, namely by:

$$\sum_{n=1}^{N} \log(m(i(g), g_n)) = 0 \in \mathfrak{g} \ . \tag{3}$$

The above equation generalizes the characteristic equation of [7] in that equation (3) was derived regardless of the specific structure of the Lie group at hand.

It is important to underline that, according to equation (2), the computation of a mean value depends on the algebraic structure as well as the geometric structure that the Lie group of interest is endowed with.

3 Specific Results Tailored to the Group of Multidimensional Rotations

In the present section, the above general-purpose results are tailored to the case of the group of multi-dimensional rotations.

3.1 Closed-Form Solution of the Characteristic Equation for the Group $SO(p)$

We are now ready to study the problem of expressing a closed form solution for the equation (2) for the group $SO(p)$.

The Lie group of multidimensional rotations is defined as:

$$SO(p) \stackrel{\text{def}}{=} \{g \in \mathbb{R}^{p\times p} | g^T g = e,\ \det(g) = 1\}\ , \tag{4}$$

where symbol ^{T} denotes matrix transpose and symbol e denotes the $p \times p$ identity matrix. The Lie algebra associated to the group $SO(p)$ is $\mathfrak{so}(p) \stackrel{\text{def}}{=} \{s \in \mathbb{R}^{p\times p} | s^T = -s\}$.

In order to define a geometric structure for the space $SO(p)$, it is necessary to describe a metric structure. In this paper, the metric structure is assumed as the one induced by the Euclidean manifold $\mathbb{R}^{p\times p}$, namely, the standard bi-invariant uniform metric. It is known that, in this case, the exponential map at identity coincides with matrix exponential.

The standard algebraic structure considered for the group $SO(p)$ is the one deriving from the general linear group of matrices, namely $(G, \cdot, ^T, e)$, where symbol $\cdot$ denotes standard matrix-to-matrix multiplication. In fact, it is clear that if $g_1, g_2 \in SO(p)$, then $g_1 \cdot g_2 \in G$ and $g_1^{-1} = g_1^T$. If such standard group structure is made use of in the equation (2), then such equation cannot be solved in closed form and should be solved for $g \in SO(p)$ by an iterative algorithm.

An alternative group-structure for the space $SO(p)$ is considered as follows. It is based on the concept of 'logarithmic multiplication' recalled in [11]:

- *Group multiplication*: $m(g_1, g_2) \stackrel{\text{def}}{=} \exp(\log g_1 + \log g_2)$, with $g_1, g_2 \in G$.
- *Group identity element*: $e =$ identity matrix in $SO(p)$.
- *Group inverse*: $i(g) = g^{-1}$, with $g \in G$.

It is easy to verify that the proposed instances of m, i, e satisfy the group axioms in $SO(p)$. Also, the logarithmic multiplication on $SO(p)$ is locally compatible with its structure of smooth manifold, as $(g_1, g_2) \to m(g_1, i(g_2)) = \exp(\log(g_1) - \log(g_2))$ is a smooth map whenever the map log may be computed.

The exponential of a matrix $s \in \mathfrak{so}(p)$ is defined by the series:

$$\exp(s) \stackrel{\text{def}}{=} \sum_{k=0}^{\infty} \frac{s^k}{k!}\ .$$

It is worth recalling that, given matrices $s_1, s_2 \in \mathfrak{so}(p)$, it holds $\exp(s_1 + s_2) \neq \exp(s_1)\exp(s_2)$ unless matrices s_1 and s_2 commute. Logarithms of a matrix $g \in SO(p)$ are solutions of the matrix equation $\exp(s) = g$. When the matrix g does not possess any negative eigenvalues, then there exists its unique real logarithm, termed 'principal logarithm', which is denoted by $\log g$. In general, given matrices $g_1, g_2 \in SO(p)$, it holds $\log(g_1 g_2) \neq \log(g_1) + \log(g_2)$. (It is worth noting that special orthogonal matrices have all eigenvalues (real and complex-valued, in general) with unitary modulus, therefore, the only case for which a

multidimensional rotation does not possess a principal logarithm is when it has one or more eigenvalue equal to -1.)

We may now apply the above structure by plugging the group multiplication and inverse operators as defined above into the equation (3). First, note that:

$$\log(m(i(g), g_n)) = \log g_n - \log g \ .$$

Now, according to equation (3), the mean value $g \in G$ must satisfy:

$$0 = \sum_{n=1}^{N} (\log g - \log g_n) = N \log g - \sum_{n=1}^{N} \log g_n \ .$$

Simple calculations lead thus to the closed-form solution:

$$g = \exp\left(\frac{1}{N} \sum_{n=1}^{N} \log g_n \right) . \tag{5}$$

The computability of the closed-form solution (5) depends on the way the principal matrix logarithm is actually calculated. For instance, if for any given matrix norm $\|\cdot\|$ it holds $\|g - e\| < 1$, then:

$$\log g = -\sum_{k=1}^{\infty} \frac{(g-e)^k}{k(-1)^k} \ .$$

For a recent discussion about the calculation of matrix logarithm (and a closed-form solution that requires different existence hypotheses), readers might see, e.g., [1].

3.2 Pushforward Map Associated to the Used Inverse Left-Translation

For completeness, we present a short derivation of the pushforward map associated to the inverse left-translation in $SO(p)$ relative to the algebraic structure described in subsection 3.1. In particular, we are interested in the pushforward map at the identity of the Lie group $SO(p)$.

Let us first recall the concept of pushforward maps for smooth manifolds and how they can be computed. Let M and H be two smooth manifolds and $\varphi : M \to H$ be a smooth map between these smooth manifolds. Given a point $x \in M$, it is defined an application $\varphi_\star$, termed *pushfoward*, as the linear map:

$$(\varphi_\star)_x : T_x M \to T_{\varphi(x)} H \ . \tag{6}$$

It can be viewed as generalization of the total derivative of ordinary calculus and can be used to push-forward tangent vectors on the manifold M to tangent vectors on the manifold H. The pushforward map may be computed via differentiation over smooth curves on the manifold M. Let $c : [-a \ a] \to M$, with

$a > 0$, be a smooth curve such that $c(0) = x$ and $\dot{c}(0) = v \in T_xM$. Then, the vector $w \in T_{\varphi(x)}H$ corresponding to the vector $v \in T_xM$ under the action of the pushforward map $(\varphi_\star)_x$ may be computed as:

$$w = \left.\frac{d\varphi(c(t))}{dt}\right|_{t=0} . \tag{7}$$

Note that the value $w \in T_{\varphi(x)}H$ depends on both values $x \in M$ and $v \in T_xM$, so we shall make use of the notation $w = (\varphi_\star)_x(v)$. The construction of a pushforward map is independent of the chosen curve $c(t)$.

Now, let us consider the case that $M = H = SO(p)$ and $\varphi \overset{\text{def}}{=} \ell_g^{-1}$ for a $g \in SO(p)$. We know that, with the group structure defined in the subsection 3.1, for $h \in SO(p)$, it holds:

$$\ell_g^{-1}(h) = m(g, h) = \exp(\log g + \log h) . \tag{8}$$

Now, let us consider a smooth curve $c(t)$ such that $c(0) = e \in SO(p)$ and then $\dot{c}(0) = v \in \mathfrak{so}(p)$. In this case it is easy to see that the application φ maps a point from a neighborhood of the identity to a point in the neighborhood of the point g, hence the pushforward map $\varphi_\star$ maps a point from the Lie algebra $\mathfrak{so}(p)$ to the tangent space $T_gSO(p)$. By definition of pushforward map we have:

$$\begin{aligned}(\varphi_\star)_e(v) &= \left.\frac{d}{dt}\exp(\log c(t) + \log g)\right|_{t=0} \\ &= (\exp_\star)_{\log c(t)+\log g}((\log_\star)_{c(t)}(\dot{c}(t)))\big|_{t=0} \\ &= (\exp_\star)_{\log(g)}((\log_\star)_e(v)) .\end{aligned} \tag{9}$$

Now, it is known that $(\log_\star)_e(v) = v$, thus, in conclusion:

$$(\ell_{g\star}^{-1})_e(v) = (\exp_\star)_{\log(g)}(v) . \tag{10}$$

The value $(\exp_\star)_u(v)$, for any $u, v \in \mathfrak{so}(p)$, may be computed via the series:

$$(\exp_\star)_u(v) = \sum_{k=1}^{\infty} \frac{1}{k!} \sum_{r=1}^{k} u^{r-1} v u^{k-r} . \tag{11}$$

The pushforward map (10) may be used to push a vector from the Lie algebra of the special orthogonal group to the tangent space of the group at any point. This expression may be useful in the numerical simulations when it is necessary to generate random tangent vectors at some point of the manifold and, more generally, to translate statistical properties from the Lie algebra to any tangent space of the manifold.

4 Conclusion

Computing averages over the Lie group of special orthogonal matrices has several applications. The common formulation in terms of Riemannian mean leads to a

matrix-type non-linear problem which is hard to solve and that usually may be tackled only via iterative algorithms.

The aim of the present paper was to propose a formulation of the problem of averaging matrices over the curved manifold-group of multidimensional rotations that gives rise to a closed-form solution.

We first presented a quite general formulation of averaging that depends on the algebraic as well as on the geometric structure that a Lie group is endowed with. We then customized the above theory to the case of the Lie group $SO(p)$ and showed that, under an appropriate choice of an algebraic/geometric structure, it is possible to obtain a closed-form solution for the mean-matrix.

It is important to underline that a closed form solution has the advantage of easiness of implementation. However, if compared to a solution obtained via a Riemannian mean theory, it has the drawback of being sub-optimal, in general. In fact, an iterative algorithm might provide a mean value that corresponds to a lesser variance [5].

References

1. Cardoso, J.A.R.: An Explicit Formula for the Matrix Logarithm. South African Optometrist 64, 80–83 (2005)
2. Ferreira, R., Xavier, J.: Hessian of the Riemannian Squared-Distance Function on Connected Locally Symmetric Spaces with Applications. In: Proc. of the 7th Portuguese Conference on Automatic Control (Controlo 2006), Special session on control, optimization and computation (2006)
3. Fiori, S.: A Theory for Learning by Weight Flow on Stiefel-Grassman Manifold. Neural Computation 13, 1625–1647 (2001)
4. Fiori, S.: Quasi-Geodesic Neural Learning Algorithms Over the Orthogonal Group: A Tutorial. Journal of Machine Learning Research 6, 743–781 (2005)
5. Fiori, S., Tanaka, T.: An Algorithm to Compute Averages on Matrix Lie Groups. IEEE Trans. on Signal Processing 57, 4734–4743 (2009)
6. Manning, R.S., Bulman, G.B.: Stability of an Elastic Rod Buckling into a Soft Wall. Proc. of the Royal Society A: Mathematical, Physical and Engineering Sciences 461, 2423–2450 (2005)
7. Moakher, M.: Means and Averaging in the Group of Rotations. SIAM Journal of Matrix Analysis and Applications 24, 1–16 (2002)
8. Prentice, M.J.: Fitting Smooth Paths to Rotation Data. The Journal of the Royal Statistical Society Series C (Applied Statistics) 36, 325–331 (1987)
9. Rao, R.P.N., Ruderman, D.L.: Learning Lie Groups for Invariant Visual Perception. Advances in Neural Information Processing Systems (NIPS) 11, 810–816 (1999)
10. Spivak, M.: A Comprehensive Introduction to Differential Geometry, 2nd edn., vol. 1. Perish Press, Berkeley (1979)
11. Warmuth, M.K.: A Bayes Rule for Density Matrices. In: Advances in Neural Information Processing Systems (NIPS 2005), vol. 18. MIT Press, Cambridge (2005)

A Lower Order Discrete-Time Recurrent Neural Network for Solving High Order Quadratic Problems with Equality Constraints

Wudai Liao, Jiangfeng Wang, and Junyan Wang

School of Electrical and Information Engineering
Zhongyuan University of Technology
Zhongyuan West Road 41, Zhengzhou, Henan Province, China

Abstract. A lower order discrete-time recurrent neural network is presented in this paper for solving higher quadratic programming. It bases on the orthogonal decomposition method and solves high order quadratic programs, especially for the case that the number of decision variables is close to the number of its constraints. The proposed recurrent neural network is globally exponential stability and converges to the optimal solutions of the higher quadratic programming. The condition for the neural network to globally converge to the optimal solution of the quadratic program is given. An illustrative example and the simulation results are presented to illustrate its performance.

Keywords: Discrete recurrent neural network, Quadratic programming, Orthogonal decomposition of matrix, Exponential stability.

1 Introduction

A quadratic programming problem with equality constraints is described as follows:

$$\begin{aligned} &\text{minimize} \quad q(x) = \frac{1}{2}x^{\mathrm{T}}Gx + g^{\mathrm{T}}x \ , \\ &\text{subject to} \quad A^{\mathrm{T}}x = b \ . \end{aligned} \tag{1}$$

Where the objective function is quadratic and the constraints are linear, $x \in \mathbf{R^n}$ is the decision variables, $G \in \mathbf{R^{n \times n}}$ is the matrix of objective coefficients. We assume the matrix G be symmetrical but not be positive definite. $g \in \mathbf{R^n}$ is the column vector of objective coefficients. $A \in \mathbf{R^{n \times m}}$ $(m < n)$, each column of A is formed by the coefficients of the corresponding constraint. We assume that the matrix A be full rank, $b \in \mathbf{R}$ is the vector of constraint coefficients.

In recent years, neural networks have been used for solving optimization problems widely, e.g., References [2,3,4]. Because of its abilities of parallel and distributed computation, neural networks fit for solving real time problems. Reference [5] had presented a continuous-time recurrent neural network developed from the orthogonal decomposition method and solved high order quadratic

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 193–198, 2010.

programs, like (1). The recurrent neural network proposed is simple in structure, and is more stable and more accuracy for solving higher quadratic programming than some existed conclusions, especially for the case that the number of decision variables is close to the number of the constraints.

However, in view of the availability of the digital hardware and the compatibility to the digital computers, the discrete-time version is more desirable in practical applications[6]. In this paper, we will propose a discrete-time network, which is the counterpart of the continue-time network in Ref[5] and give the condition for the discrete-time recurrent neural network with global convergence.

2 Network Descriptions

In this section, we will firstly review the continuous-time network introduced in Ref[5], which is represented by a system of differential equation. Then, we propose its discrete-time version by discretization, which is expressed by a system of difference equations.

2.1 Continuous-Time Network

For (1), we select two matrices $S \in \mathbf{R^{n\times m}}$ and $C \in \mathbf{R^{n\times(n-m)}}$, such that $A^{\mathrm{T}}S = I_m$, $A^{\mathrm{T}}C = 0$ and $(S : C) \in \mathbf{R^{n\times n}}$ is nonsingular, I_m is m-dimensional unit matrix. The column vectors $c_1, c_2, \cdots, c_{n-m}$ of the matrix C is the base vector group of the linear space g. For any feasible point $N(A) = \{\delta \in \mathbf{R^n} : \mathbf{A}^{\mathrm{T}}\delta = \mathbf{0}\}$, we have $A^{\mathrm{T}}x = b$. The constraint equation has a particular solution Sb , and its general solutions have the form: $x = Sb + \delta$, $\delta \in N(A)$. So, we choose the linear transformation:

$$x = Cy + Sb, \quad y \in \mathbf{R^{n-m}} \ . \tag{2}$$

Replacing x in (1) by (2), we have the following lower order unconstrained optimization problems:

$$\begin{aligned} \min \ \ q(x) = \phi(y) = &\frac{1}{2}y^{\mathrm{T}}(C^{\mathrm{T}}GC)y + (g + GSb)^{\mathrm{T}}Cy \\ &+ \frac{1}{2}(2g + GSb)^{\mathrm{T}}Sb \ . \end{aligned} \tag{3}$$

Obviously, we have the following lemma.

Lemma 1. *x^* is the global minimum point of (1) if and only if there exists a matrix C, such that the matrix $C^{\mathrm{T}}GC$ is positive definite.*

By the orthogonal decomposition method, the QR decomposition of the matrix A has the form:

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix} = (\, Q_1 \quad Q_2 \,)\begin{pmatrix} R \\ 0 \end{pmatrix} = Q_1 R \ .$$

Where, $Q \in \mathbf{R^{n\times n}}$ is an orthogonal matrix, $R = (r_{ij}) \in \mathbf{R^{m\times m}}$ is an upper triangular matrix with $r_{ii} > 0$, $Q_1 \in \mathbf{R^{n\times m}}$ and $Q_2 \in \mathbf{R^{n\times(n-m)}}$. It is easy to

verify that $S = Q_1 R^{-\mathrm{T}}$; $C = Q_2$ satisfy the properties, i.e. $A^{\mathrm{T}} S = I_m$, $A^{\mathrm{T}} C = 0$ and $(S : C)$ is nonsingular.

For (3), by setting the gradients of $\phi(y)$ to zero, the necessary condition gives rise to the following matrix form algebraic equation:

$$\left(C^{\mathrm{T}} G C\right) y = -C^{\mathrm{T}} \left(g + GSb\right) \quad . \tag{4}$$

In order to get the real time solution of (4), it can be realized by the analogue circuits of the recurrent neural network. The dynamical equations of the recurrent neural network is as follows:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = Wy + \theta \quad . \tag{5}$$

Where $W = -C^{\mathrm{T}} G C$ is the $(n-m)$-dimensional connection weight matrix, and $\theta = -C^{\mathrm{T}} \left(g + GSb\right)$ is the $(n-m)$-dimensional biasing threshold vector.

2.2 Discrete-Time Network

By taking discretization of the dynamical equation of the continuous-time network (5), we propose the dynamical equation for its discrete-time counterpart as

$$y(k+1) = y(k) + \alpha(Wy(k) + \theta) = (I + \alpha W)y(k) + \alpha\theta \quad . \tag{6}$$

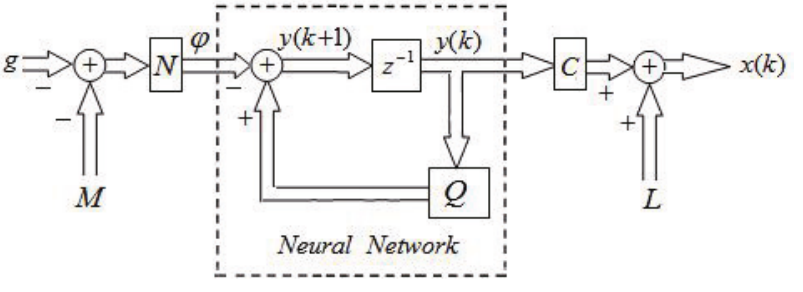


Fig. 1. Block diagram for the configuration of the discrete-time network

Fig 1 shows the block diagram for the configuration of the discrete-time network.Where $x(k) = Cy(k) + Sb$, α is a positive fixed step size and I is an identity matrix. $Q = I + \alpha W$, $\varphi = -\alpha\theta$, $M = GL$, $N = \alpha C^{\mathrm{T}}$ and $L = Sb$.

3 Convergence Analysis

Lemma 2. *The equilibrium point of the dynamical system (6) is equal to the optimal solution of the quadratic program (1) .*

Proof. Let y^* be the equilibrium point of the dynamical system (6) , then by the nature of equilibrium point, we have

$$y^* = (I + \alpha W)y^* + \alpha\theta \ . \tag{7}$$

Equation (7) implies

$$Wy^* + \theta = 0 \ , \tag{8}$$

which satisfies the optimality conditions (4). Hence the equilibrium point of the dynamical system (6) maps to the optimal solution of the quadratic program (1).

Theorem 1. *The discrete-time network (6) is asymptotically convergent to the optimal solution of the quadratic program (1) if*

$$\alpha < \min_{1 \le i \le n-m} \left\{ \frac{-2Re[\rho_i(W)]}{|\rho_i(W)|^2} \right\} . \tag{9}$$

Where $\rho_i(W)$ denotes the ith eigenvalue of the matrix W, $Re[\rho_i(W)]$ and $|\rho_i(W)|$ are the real part and the absolute value of $\rho_i(W)$, respectively.

Proof. The discrete-time network (6) is described by first-order difference equation. So, it is a linear system. By the linear system theory, we know that system (8) is asymptotically stable if all eigenvalues of the matrix $(I+\alpha W)$ have absolute values less than 1, i.e.,

$$|\rho_i(I + \alpha W)| < 1, \forall i = 1, 2, \cdots, n - m \ . \tag{10}$$

By using the properties of eigenvalue of matrices, the inequality (10) can be written as

$$|\rho_i(I) + \alpha\rho_i(W)| < 1, \forall i = 1, 2, \cdots, n - m \ . \tag{11}$$

Since for all $\rho_i(I) = 1$, and let $\rho_i(W) = Re[\rho_i(W)] + jIm[\rho_i(W)]$, where $j = \sqrt{-1}$. For all $i = 1, 2, \cdots, n - m$, inequality (11) becomes

$$|1 + \alpha Re[\rho_i(W)] + j\alpha Im[\rho_i(W)]| < 1 \ . \tag{12}$$

So, we can get

$$\alpha < \frac{-2Re[\rho_i(W)]}{|\rho_i(W)|^2}, \forall i = 1, 2, \cdots, n - m \ . \tag{13}$$

The desired result of the theorem is obtained. The proof is complete.

4 Simulation Results

Considering the quadratic program (1) with

$$G = \begin{pmatrix} 6 & 0 & 1 & -2 \\ 0 & 4 & 2 & 0 \\ 1 & 2 & 5 & 1 \\ -2 & 0 & 1 & 6 \end{pmatrix}, g = \begin{pmatrix} -17 \\ 18 \\ 9 \\ 13 \end{pmatrix},$$

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 6 \end{pmatrix}, \; b = \begin{pmatrix} 16 \\ 7 \end{pmatrix}.$$

The theoretical optimal solution for this quadratic program is

$$x^{\mathrm{T}} = (\, 7.1615 \quad -0.7975 \quad -3.839 \quad 3.2375 \,).$$

By using the orthogonal decomposition method,and according to the equation (4) and (5) , we have

$$W = \begin{pmatrix} -3.5276 & 0.8624 \\ 0.8624 & -7.1151 \end{pmatrix}, \; \theta = \begin{pmatrix} 6.1155 \\ 5.7626 \end{pmatrix}.$$

According to the convergence condition (9), we can obtain

$$\alpha < \min_{1 \le i \le n-m} \left\{ \frac{-2Re[\rho_i(W)]}{|\rho_i(W)|^2} \right\} = 0.2735 \; .$$

So, the discrete-time network is globally convergent to the optimal solution of the desired quadratic program if we select $\alpha < 0.2735$.

Fig 2 shows the time evolution of the decision variable generated by the discrete-time network with different values of the fixed step size α. In subplot (a), we select the fixed step size to be $\alpha = 0.09$. It notes that the discrete-time network generates the optimal solution $x^{\mathrm{T}} = (\, 7.1613 \quad -0.7971 \quad -3.835 \quad 3.237 \,)$ after about 30 iterations, which equals to the theoretical optimal solution. In subplot (b), where the fixed step size is selected to be $\alpha = 0.22$, it can achieve

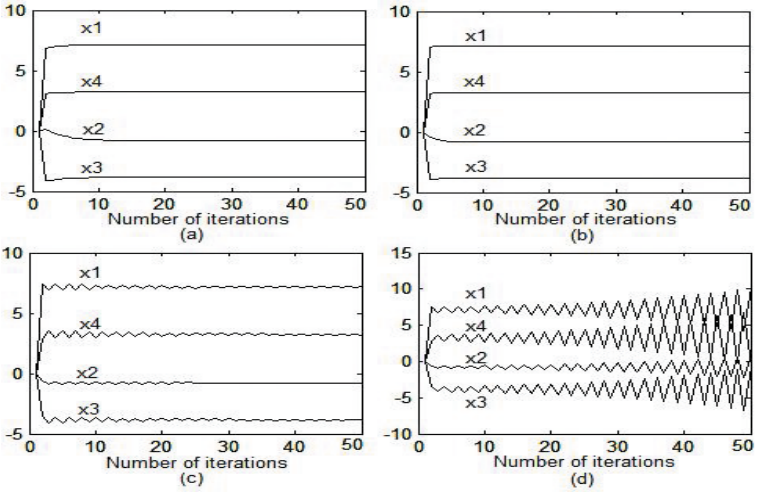


Fig. 2. Time evolution of the decision variable generated by the discrete-time network with different fixed steps. Transient states of the decision variable with (a) $\alpha = 0.09$; (b) $\alpha = 0.22$; (c) $\alpha = 0.27$; (d) $\alpha = 0.28$.

to the theoretical optimal solution only after about 12 iterations, and the convergence process of the system shows to be very steady. In subplot (c), when the fixed step size is set as $\alpha = 0.27$, which is close to the upper bound but under it, we find the estimated optimal solution after 50 iterations can not be equal to the theoretical optimal solution, and the convergence process of the system shows to be unsteady. Subplot (d) shows the network is divergent when the fixed step size $\alpha = 0.27$, which is greater than its upper bound 0.2735. Therefore, if we want to speed up the rate of convergence and get a steady network, we can select a greater step size below its upper bound, but not too close to it.

5 Concluding Remarks

This paper presents a lower order discrete-time network for solving high order convex quadratic programs. We have given the condition of the fixed step size used in the discrete-time model, which guarantees the proposed discrete-time network to be asymptotically convergent to the optimal solution of the desired quadratic program. This discrete-time model is easier to be implemented in practical applications because of the availability of the digital hardware. Realized in dedicated digital hardware such as field-programmable gate arrays (FPGA) , the discrete-time recurrent neural network is suitable for many applications where intensive computation is essential such as robot control and filter design.

Acknowlegement

The work is supported by National Natural Science Foundation of China (60774051) and National Natural Science Foundation of Henan Province (072300410400).

References

1. Fletcher, M.: Practical Methods of Optimization. John Wiley & Sons, Chichester (1981)
2. Hopfield, J.J., Tank, D.W.: Neural Computation of Decisions in Optimal Problems. Biological Cybernetics 52(3), 141–152 (1985)
3. Kennedy, M., Chua, L.O.: Neural Networks for Nonlinear Programming. IEEE Trans. CAS 35(5), 554–562 (1988)
4. Wang, J.: Recurrent Neural Network for Solving Quadtratic Propramming Problems with Equality Constraints. Electronics Letter 28(14), 1345–1347 (1992)
5. Liao, W.D., Wang, J.F.: A Lower Order Recurrent Neural Network for Solving Higher Order Quadratic Programming Problems with Equality Constraints. In: Proceedings of the Second International Joint Conference on Computational Sciences, pp. 176–178 (2009)
6. Tang, W.S., Wang, J.: A Discrete-time Lagrange Network for Solving Constrained Quadratic Programs. International Journal of Neural Systems 10(4), 261–265 (2000)

A Experimental Study on Space Search Algorithm in ANFIS-Based Fuzzy Models

Wei Huang[1], Lixin Ding[1], and Sung-Kwun Oh[2]

[1] State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China
[2] Department of Electrical Engineering, The University of Suwon, San 2-2, Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
ohsk@suwon.ac.kr

Abstract. In this study, we propose a space search algorithm (SSA) and then introduce a hybrid optimization of ANFIS-based fuzzy models based on SSA and information granulation (IG). The overall hybrid identification of ANFIS-based fuzzy models comes in the form of two optimization mechanisms: structure identification (such as the number of input variables to be used, a specific subset of input variables, the number of membership functions, and polynomial type) and parameter identification (viz. the apexes of membership function). The structure identification is developed by SSA and C-Means while the parameter estimation is realized via SSA and a standard least square method. The evaluation of the performance of the proposed model was carried out by using two representative numerical examples such as gas furnace, and Mackey-Glass time series. A comparative study of SSA and PSO demonstrates that SSA leads to improved performance both in terms of the quality of the model and the computing time required. The proposed model is also contrasted with the quality of some "conventional" fuzzy models already encountered in the literature.

Keywords: Space search algorithm, Particle swarm algorithm, Information granulation, ANFIS-based fuzzy inference system.

1 Introduction

In recent years, fuzzy modeling has been utilized in many fields for engineering, medical engineering, and even social science [1]. As for fuzzy model construction, identification of fuzzy rules is one of most important parts in the design of rule-based fuzzy modeling.

Many identification methods for fuzzy models have been studied over the past decades. In the early 1980s, linguistic modeling [2] was proposed as primordial identification methods for fuzzy models. Then Tong et.al [3], C.W.Xu et.al [4] studied different approaches for fuzzy models. While appealing with respect to the basic topology (a modular fuzzy model composed of a series of rules) [5] , these models still await formal solutions as far as the structure optimization of the model is concerned, say a construction of the underlying fuzzy sets – information granules being viewed as basic building blocks of any fuzzy model. Oh and Pedrycz [6] have proposed some enhancements to the model, yet the problem of finding "good" initial parameters of

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 199–206, 2010.

the fuzzy set in the rules remains open. To solve this problem, several genetically identification methods for fuzzy models have been proposed. Liu et.al [7], Chung and Kim [8] and others have discussed employing genetic algorithms to fuzzy models, respectively. In a word, evolutionary identification methods have proven to be useful in optimization of such problems.

In this study, we propose a space search algorithm (SSA) and then introduce a hybrid optimization of ANFIS-based fuzzy models based on SSA and information granulation (IG). SSA is exploited here to carry out the parameter estimation of the fuzzy models as well as to realize structure optimization. The identification process is comprised of two phases, namely a structure optimization (the number of input variables to be used, a specific subset of input variables, and the number of membership functions) and parametric optimization (apexes of membership function). To evalutate the performance of the proposed model, we exploit two kinds of well-known data set. A hybrid optimization of ANFIS-based fuzzy models based on PSO and IG is also implemented for the comparative study.

2 IG-Based Fuzzy Model

Granulation of information is an inherent and omnipresent activity of human beings carried out with intent of gaining a better insight into a problem under consideration and arriving at its efficient solution. In particular, granulation of information is aimed at transforming the problem at hand into several smaller and therefore more manageable tasks. The identification of the conclusion parts of the rules deals with a selection of their structure (type 1, type 2, type 3 and type 4) that is followed by the determination of the respective parameters of the local functions occurring there. The conclusion part of the rule that is extended form of a typical fuzzy rule in the TSK (Takagi-Sugeno-Kang) fuzzy model has the form.

$$R^j : If\ x_1\ is\ A_{1c}\ and \cdots and\ x_k\ is\ A_{kc}\ then\ y_j - M_j = f_j(x_1, \cdots, x_k) \tag{1}$$

The calculations of the numeric output of the model, based on the activation (matching) levels of the rules there, rely on the following expression.

$$y^* = \frac{\sum_{j=1}^{n} w_{ji} y_i}{\sum_{j=1}^{n} w_{ji}} = \frac{\sum_{j=1}^{n} w_{ji}(f_j(x_1, \cdots, x_k) + M_j)}{\sum_{j=1}^{n} w_{ji}} = \sum_{j=1}^{n} \hat{w}_{ji}(f_j(x_1, \cdots, x_k) + M_j) \tag{2}$$

Here, as the normalized value of w_{ji}, we use an abbreviated notation to describe an activation level of rule R^j to be in the form

$$\hat{w}_{ji} = \frac{w_{ji}}{\sum_{j=1}^{n} w_{ji}},\ \hat{w}_{ji} = \frac{A_{j1}(x_{1i}) \times \cdots \times A_{jk}(x_{ki})}{\sum_{j=1}^{n} A_{j1}(x_{1i}) \times \cdots \times A_{jk}(x_{ki})} \tag{3}$$

where, R^j is the j-th fuzzy rule, x_k represents the input variables, A_{kc} is a membership function of fuzzy sets, a_{jk} is a constant, V_{jk} and M_j is a center value of the input and output data, respectively, n is the number of fuzzy rules, y^* is the inferred output value, w_{ji} is the premise fitness matching R^j (activation level).

We use two performance indexes as the standard root mean squared error (RMSE) and mean squared error (MSE)

$$PI(or\ E\_PI) = \begin{cases} \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - y_i^*)^2}, & (RMSE) \\ \frac{1}{m}\sum_{i=1}^{m}(y_i - y_i^*)^2. & (MSE) \end{cases} \tag{4}$$

where y^* is the output of the fuzzy model, m is the total number of data, and i is the data number.

The consequence parameters a_{jk} can be determined by the standard least-squares method that leads to the expression

$$\hat{\mathbf{a}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{5}$$

3 Optimization

3.1 Space Search Evolutionary Algorithm

SSA is an evolutionary algorithm whose search method comes with the analysis of the solution space. In essence, the solution space is the set of all feasible solutions for the optimization problem (or mathematical programming problem), which is stated as the problem of determining the best solution coming from the solution space. The search method is based on the operator of space search, which generates two basic steps: generate new space and search new space. Search in the new space is realized by randomly generating a new solution (individual) located in this space. The role of space search is to create new solutions from old ones. Regarding the generation of the new space, we consider two cases: (a) space search based on M selected solutions (denoted here as Case I), and (b) space search based on one selected solution (Case II).

The features of the SSA are highlighted as follows.

(1) The SSA leads to better performance when finding global optimization than PSO, especially in the optimization problems with larger solution spaces. SSA seach the same size of solution space as PSO. However, SSA search the solutions based on the relative adequate analyzing space (search adjacent space in Case I and search long distance space in Case II) while PSO search the solutions without such adequate analyzing space.

(2) The SSA leads to shorter computing time when being compared with the conventional PSO. Each solution is updated in PSO while SSA generates only two new solutions each generations. That is, in one generation, individuals which correspond to lots of new solutions are evaluated in PSO while only two new solutions (individual) are evaluated in SSA. This operation procedure enables us to carry out the rapid CPU operation for hybrid identification of fuzzy systems.

3.2 Particle Swarm Optimization

PSO is an example of a modern search heuristics belonging to the category of Swarm Intelligence methods. PSO involves two competing search strategy aspects [9]. One deals with a social facet of the search; according to this, individuals ignore their own experience and adjust their behavior according to the successful beliefs of individuals occurring in their neighborhood. The cognition aspect of the search underlines the importance of the individual experience where the element of population is focused on its own history of performance and makes adjustments accordingly. Unlike many other heuristic techniques, PSO has a flexible and well-balanced mechanism to enhance the global and local exploration abilities.

3.3 Hybrid Optimization of ANFIS-Based Fuzzy Models

The standard gradient-based optimization techniques might not be effective in the context of rule based systems given their nonlinear character (in particular the form of the membership functions) and modularity of the systems. This suggests us to explore other optimization techniques. Figure 1 depicts the arrangement of chromosomes commonly used in fuzzy modeling [6,10]. Genes for structure optimization are separated from genes used for parameter optimization. The size of the chromosomes for structure identification of the IG-based fuzzy model is determined according to the number of all input variables of the system. The size of the chromosomes for parameter identification depends on structurally optimized ANFIS-based fuzzy inference system.

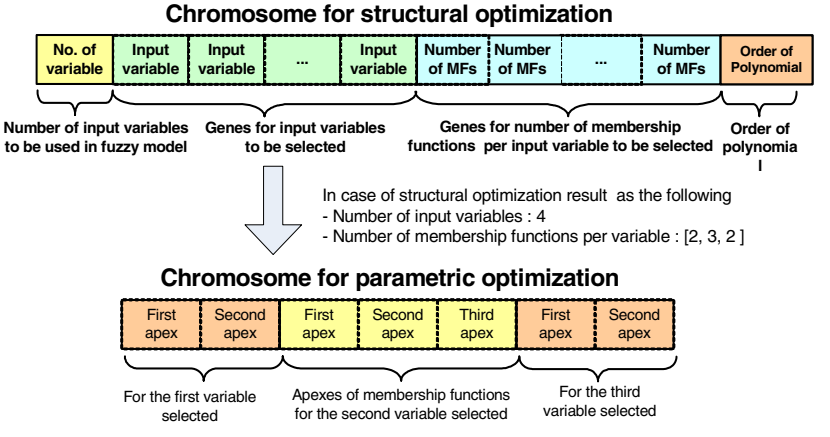


Fig. 1. Arrangement of chromosomes for the optimization of fuzzy model

The objective function (performance index) is a basic mechanism guiding the evolutionary search carried out in the solution space. The objective function includes both the training data and testing data and comes as a convex combination of the two components.

$$f(PI,\ E\_PI) = \theta \times PI + (1 - \theta) \times E\_PI \tag{6}$$

4 Experimental Studies

This section includes comprehensive numeric studies illustrating the design of the fuzzy model. We use two well-known data sets. PI denotes the performance index for training data and E_PI for testing data. The following values of the parameters are used: maximum number of generations is 150; maximal velocity, v_{max}, is 20% of the range of the corresponding variables; w=0.4 and acceleration constants c_1 and c_2 are set to 2.0. The maximal velocity was set to 0.2 for the search carried out in the range of the unit interval [0,1]. The algorithm terminatesafter running 1000 generations. The parameters of SSA are as follows. We use 150 generations and a size of 100 populations (individuals) for structure identification and run the method for 1,000 generations. The population size is 60 for parameter identification. In each generation, we first search the space based on 8 solutions generated randomly and then search the space based on the best solution.

4.1 Gas Furnace Process

The second well-known dataset is time series data of a gas furnace utilized by Box and Jenkins [2-6,10]. The time series data is comprised of 296 input-output pairs resulting from the gas furnace process has been intensively studied in the previous literature. The delayed terms of methane gas flow rate $u(t)$ and carbon dioxide density $y(t)$ are used as six input variables with vector formats such as $[u(t-3), u(t-2), u(t-1), y(t-3), y(t-2), y(t-1)]$. $y(t)$ is used as output variable. The first 148 pairs are used as the training data while the remaining 148 pairs are the testing data set for assessing the predictive performance. MSE is considered as a performance index.

The results show that SSA has less identification error, less CPU time and rapid convergence in comparison with PSO. The identification error of the proposed model

Table 1. Comparative analysis of selected models (GAS)

Model		PI_t	PI	E_PI	No. of rules
Tong's model[3]		0.469			19
Pedrycz's model[2]		0.776			20
Xu's model[4]		0.328			25
Sugeno's model[5]		0.355			6
Oh et al.'s model[6]	Simplified		0.024	0.328	4
	Linear		0.022	0.326	4
			0.021	0.364	6
HCM+GA [10]	Simplified		0.035	0.289	4
			0.022	0.333	6
	Linear		0.026	0.272	4
			0.020	0.264	6
Our model	PSO+IG		0.019	0.284	4
			0.015	0.273	6
	SSA+IG		0.017	0.266	4
			0.015	0.260	6

is compared with the performance of some other models; refer to Table 1. It is easy to see that the proposed model outperforms several previous fuzzy models known in the literature.

4.2 Chaotic Mackey-Glass Time Series

A chaotic time series is generated by the chaotic Mackey–Glass differential delay equation [11-16] of the form:

$$\dot{x}(t) = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t)$$

The prediction of future values of this series arises is a benchmark problem that has been used and reported by a number of researchers. From the Mackey–Glass time series x(t), we extracted 1000 input–output data pairs for the type from the following the type of vector format such as: [x(t-30), x(t-24), x(t-18), x(t-12), x(t-6), x(t); x(t +6)] where t = 118–1117. The first 500 pairs were used as the training data set for IG-based FIS while the remaining 500 pairs were the testing data set for assessing the predictive performance. To come up with a quantitative evaluation of the fuzzy model, we use the standard RMSE performance index as like Eq. (4). The results show that SSA has better performance index, less CPU time and rapid convergence in comparison with PSO. Table 2 summarizes the results of comparative analysis of the proposed model with respect to other constructs. Here PI_t denotes the performance

Table 2. Comparative analysis of selected models (Mackey)

Model		PI_t	PI	E_PI	NDEI	No. of rules
Support vector regression model[11]			0.023	1.028	0.0246	
Multivariate adaptive regression splines [11]			0.019	0.316	0.0389	
Standard neural networks			0.018	0.411	0.0705	15 nodes
RBF neural networks			0.015	0.313	0.0172	15 nodes
Wang's model[12]		0.004				7
		0.013				23
ANFIS [13]			0.0016	0.0015	0.007	16
FNN model[14]			0.014	0.009		
Incremental type multilevel FRS [15]			0.0240	0.0253		25
Aggregated type multilevel FRS[15]			0.0267	0.0256		36
Hierarchical TS-FS[16]			0.0120	0.0129		28
Our model	PSO+IG		0.00346	0.00323	0.0157	8
			0.00033	0.00035	0.0057	16
	SSA+IG		0.00321	0.00302	0.0155	8
			0.00010	0.00011	0.0018	16

index for total process data, the non-dimensional error index (NDEI) is defined as the RMSE divided by the standard deviation of the target series.

5 Concluding Remarks

This paper contributes to the research area of the hybrid optimization of ANFIS-based fuzzy models in the following two important aspects: 1) we proposed a space search evolutionary. From the perspective of the size of the solution space, SSA exhibits better performance in finding global optimization and less CPU time than the "conventional" PSO. 2) we introduced the hybrid optimization of ANFIS-based fuzzy models based on the SSA and information granulation. It is shown that the coding scheme introduced here leads to chromosomes which help decrease the number of unfeasible solutions arising in the process of evolutionary computing. Numerical experiments using four well-known data set show that the model constructed with the aid of the SSA exhibits better performance in comparison with the PSO-constructed fuzzy model.

Acknowledgements. This work was supported by the GRRC program of Gyeonggi province [GRRC SUWON2009-B2, Center for U-city Security & Surveillance Technology] and also supported by KESRI(2009T100100563), which is funded by MKE(Ministry Knowledge Economy).

References

1. Nataraja, M.C., Jayaram, M.A., Ravikumar, C.N.: Prediction of Early Strength of Concrete: A Fuzzy Inference System Model. J. International Journal of Physical Sciences 1, 47–56 (2006)
2. Pedrycz, W.: An Identification Algorithm in Fuzzy Relational System. J. Fuzzy Sets Syst. 13, 153–167 (1984)
3. Tong, R.M.: The Evaluation of Fuzzy Models Derived from Experimental Data. J. Fuzzy Sets Syst. 13, 1–12 (1980)
4. Xu, C.W., Zailu, Y.: Fuzzy Model Identification Self-learning for Dynamic System. J. IEEE Trans. on System, Man and Cybernetics 17, 683–689 (1987)
5. Sugeno, M., Yasukawa, T.: Linguistic Modeling Based on Numerical Data. In: IFSA 1991 Brussels, Computer, Management & System Science, pp. 264–267 (1991)
6. Oh, S.K., Pedrycz, W.: Identification of Fuzzy Systems by Means of An Auto-Tuning Algorithm and Its Application to Nonlinear Systems. J. Fuzzy Sets and Syst. 115, 205–230 (2000)
7. Liu, F., Lu, P., Pei, R.: A New Fuzzy Modeling and Identification Based on Fast-cluster and Genetic Algorithm. J. Intell. Contr. Automat. 1, 290–293 (2004)
8. Chung, W.Y., Pedrycz, W., Kim, E.T.: A New Two-phase Approach to Fuzzy Modeling for Nonlinear Function Approximation. J. IEICE Trans. Info. Syst. 9, 2473–2483 (2006)
9. Gaing, Z.L.: A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System. J. IEEE Trans. Energy Conversion 19, 384–391 (2004)
10. Park, B.J., Pedrycz, W., Oh, S.K.: Identification of Fuzzy Models with the Aid of Evolutionary Data Granulation. In: IEE Proc.-Control Theory and Applications, vol. 148, pp. 406–418 (2001)

11. Krishnaiah, P.R., Kanal, L.N. (eds.): Classification, Pattern Recognition, and Reduction of Dimensionality. Handbook of Statistics, vol. 2. North-Holland, Amsterdam (1982)
12. Wang, L.X., Mendel, J.M.: Generating Fuzzy Rules from Numerical Data with Applications. J. IEEE Trans. on System, Man and Cybernetics 22, 1414–1427 (1992)
13. Jang, J.S.R.: ANFIS: Adaptive-network-based Fuzzy Inference System. J. IEEE Trans. System Man Cybernet. 23, 665–685 (1993)
14. Maguire, L.P., Roche, B., Mcginnity, T.M., Mcdaid, L.J.: Predicting a Chaotic Time Series Using A Fuzzy Neural Network. J. Inform. Sci. 112, 125–136 (1998)
15. Duan, J.C., Chung, F.L.: Multilevel Fuzzy Relational Systems: Structure and Identification. J. Soft Comput. 6, 71–86 (2002)
16. Chen, Y., Yang, B., Abraham, A.: Automatic Design of Hierarchical Takagi-Sugeno Type Fuzzy Systems Using Evolutionary Algorithms. J. IEEE Trans. Fuzzy Systems 15, 385–397 (2007)

Optimized FCM-Based Radial Basis Function Neural Networks: A Comparative Analysis of LSE and WLSE Method

Wook-Dong Kim[1], Sung-Kwun Oh[2], and Wei Huang[3]

[1,2] Department of Electrical Engineering, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
ohsk@suwon.ac.kr
[3] State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China

Abstract. In this paper, we introduce a new architecture of optimized FCM-based Radial Basis function Neural Network by using space search algorithm and discuss its comprehensive design methodology. As the consequent part of rules of the FCM-based RBFNN model, four types of polynomials are considered. The performance of the FCM-based RBFNN model is affected by some parameters such as the number of cluster and the fuzzification coefficient of the fuzzy clustering (FCM) and the order of polynomial standing in the consequent part of rules, we are required to carry out parametric optimization of network. The space evolutionary algorithm(SEA) being applied to each receptive fields of FCM-based RBFNN leads to the selection of preferred receptive fields with specific local characteristics available within the FCM-based RBFNN. The performance of the proposed model and the comparative analysis between WLSE and LSE are illustrated with by using two kinds of representative numerical dataset.

Keywords: Radial basis function neural network, Fuzzy C-means clustering, Space evolutionary algorithm, Machine learning data.

1 Introduction

A great deal of attention in fuzzy sets has been paid to advanced techniques of system modeling and the design of fuzzy models. To highlight a few representative trends, it is essential to put the overall modeling area in some retrospect. In the early 1980s, linguistic modeling and fuzzy relation equation-based approach were proposed as primordial identification schemes of fuzzy models. The general class of Sugeno–Takagi models gave rise to more sophisticated yet more complex rule-based systems where the rules were equipped with conclusions being formed as local linear regression models. Some enhanced fuzzy models that have high-order polynomials as consequent part were presented by Bikdash [1].

Radial basis function neural networks (RBFNN) in neural network have been known to be very attractive for many research fields. An important property of RBFNN is that form a unifying link among many different research area such as

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 207–214, 2010.

function approximation, regularization, noisy interpolation and pattern recognition. In order to reduce the optimization effort, various learning techniques have been considered. Alexandridis et al. proposed a variable selection of RBF neural network using an evolutionary algorithm.

In this paper, the FCM-based RBFNN designed with the aid of FCM and the WLSE method involves structural as well as parametric optimization. As far as the structure optimization is concerned, there are three components to consider, i.e., the number of the cluster (equal to consequent rules), fuzzification coefficient used in the FCM algorithm and the order of the polynomials in the consequent parts. These three components impact the performance of the FCM-based RBFNN and need to be optimized. In this paper, we carried out the structural as well parametric optimization by means of the Space Evolutionary Algorithm (SEA).

Section 2 describes the architecture of the FCM-based RBFNN and section 3 presents a learning method applied to the construction of proposed model. Section 4 deals with the SEA and the optimization of proposed model using the SEA. Section 5 presents the experimental results. Finally, some conclusions are drawn in Section 6.

2 The Architecture of FCM-Based RBF Neural Network

The proposed FCM-based RBFNN comes as an extended structure of the conventional RBFNN and consists of the premise part and consequent part as shown in Fig. 1. The construction of the conventional RBFNN involves an input layer, a hidden layer and an output layer with feed forward architecture. The input layer denotes the number of n-dimensional input variables. All nodes of input layer connected to hidden nodes in hidden layer. The hidden layer also completely connected to output layer. The output layer, which contains the hidden node, achieves a linear combination on this new space.

The premise part of proposed model used FCM algorithm divide input space as the number of cluster. The consequent part used WLSE estimate the values of the coefficients of the polynomials.

The consequent part of rules of the FCM-based RBFNN model, four types of polynomials are considered. Those are constant, linear, quadratic and modified quadratic.

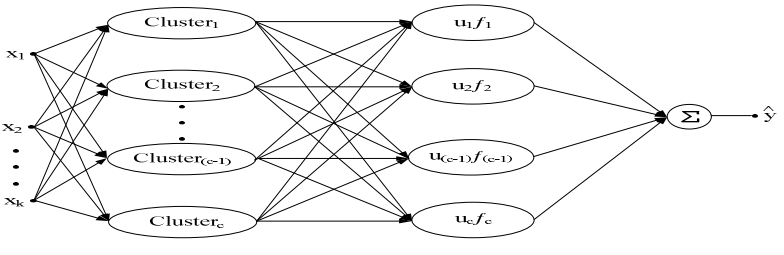


Fig. 1. Architecture of FCM-based RBFNN

3 The Learning Method of FCM-Based RBF Neural Network

3.1 The Learning of Premise Part

Bezdek introduced several clustering algorithms based on fuzzy set theory and extension of the least-squares error criterion. The most of fuzzy clustering methods are derived from Bezdeck's FCM that is a data clustering algorithm that each data point is associated with a cluster through a membership degree between 0 and 1. FCM clustering partitions collection of n data set into c fuzzy groups and finds a cluster center in each fuzzy group, such that, objective function of a dissimilarity measure is minimized.

FCM clustering algorithm as follows:

[step1] Select the number of cluster c ($2 \leq c \leq n$) and fuzzification coefficient m ($1< m < \infty$) and initialize membership matrix $U^{(0)}$ using random value 0 between 1.

Denote iteration of algorithm r(r=1, 2, 3, …).

$$\sum_{i=1}^{c} u_{ik} = 1, \forall k = 1, \cdots, n (0 < \sum_{i=1}^{c} u_{ik} < 1) \tag{1}$$

Where, n is the number of data.

[step2] Calculate the center point ($\mathbf{v_i} \mid i=1, 2, ..., c$) of each fuzzy cluster using (2).

$$v_i = \frac{\sum_{k=1}^{n} u_{ik}^{m} \mathbf{x}_{ik}}{\sum_{k=1}^{n} u_{ik}^{m}} \tag{2}$$

[step3] Calculate the distance between input variables and center points using (3)-(5)

$$J(u_{ik}, \mathbf{v}_i) = \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^m d(\mathbf{x}_k, \mathbf{v}_i)^2 \tag{3}$$

Where, the membership element $\{u_{ik}\}$ take values from the interval [0, 1] and satisfy (1), $d(\mathbf{x}_k, \mathbf{v}_i)$ is any inner product norm metric of the distance between the input vector $\mathbf{x}_k \in \mathbf{X}$ and the center point of cluster $\mathbf{v}_i \in \Re$.

$$d_{ik} = d(\mathrm{x_k}, \mathrm{v_i}) = [\sum_{j=1}^{l} (x_{ik} - v_{ij})^2]^{1/2} \qquad (1 \leq l \leq n \quad 1 \leq k \leq n \quad 1 \leq i \leq c) \tag{4}$$

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\mathbf{x}_k - \mathbf{v}_i\|}{\|\mathbf{x}_k - \mathbf{v}_j\|} \right)^{(2/(m-1))}}, \tag{5}$$

[step4] The objective function in (6) calculate new membership matrix $U^{(r+1)}$

$$u_{ik}^{r+1} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik}^{m}}{d_{jk}^{m}} \right)^{(2/(m-1))}}, \tag{6}$$

3.2 The Learning Method of Consequent Part

The learning of the consequent part is concerned with the estimation of the parameters of the polynomial of the local model. The main difference between the WLSE and its standard LSE is the weighting scheme which comes as a part of the WLSE and makes its focus on the corresponding local model. Because global learning is oriented to minimize its overall squared error between output and global model output, local models which are obtained by using LSE do not properly represent local input-output characteristics of each sub-space resulting from the division of the input space. As a result, the interpretability of the local models which are obtained by using LSE tends to be limited or non-existent.

In the WLSE, we estimate the optimal coefficients of the model through the minimization of the objective function Q_L.

$$Q_L = \sum_{i=1}^{n} \sum_{k=1}^{m} w_{ik} (y_k - f_i(\mathbf{x}_k - \mathbf{v}_i))^2 \tag{7}$$

Where, w_{ik} is the normalized firing strength (activation level) of the i^{th} rule

The performance index J_L can be re-written using matrix notation

$$\begin{aligned} Q_L &= \sum_{i=1}^{n} (\mathbf{Y} - \mathbf{X}_i \mathbf{a}_i)^T \mathbf{W}_i (\mathbf{Y} - \mathbf{X}_i \mathbf{a}_i) \\ &= \sum_{i=1}^{n} \left(\mathbf{W}_i^{1/2} \mathbf{Y} - \mathbf{W}_i^{1/2} \mathbf{X}_i \mathbf{a}_i\right)^T \left(\mathbf{W}_i^{1/2} \mathbf{Y} - \mathbf{W}_i^{1/2} \mathbf{X}_i \mathbf{a}_i\right) \end{aligned} \tag{8}$$

Where, $\mathbf{a}_i$ is the vector of coefficient of i^{th} consequent polynomial (local model), is $\mathbf{Y}$ the vector of output data, $\mathbf{W}_i$ is the diagonal matrix (weighting factor matrix) which represents degree of activation of the individual information granules by the input data. $\mathbf{X}_i$ is matrix which is formed with input data and centers point of cluster. In case the consequent polynomial is Type 2 (linear or a first-order polynomial), $\mathbf{X}_i$ and $\mathbf{a}_i$ can be expressed as follows

$$\mathbf{W}_i = \begin{bmatrix} w_{i1} & 0 & \cdots & 0 \\ 0 & w_{i2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_{in} \end{bmatrix} \qquad \mathbf{X}_i = \begin{bmatrix} 1 & x_{11} & \cdots & x_{l1} \\ 1 & x_{12} & \cdots & x_{l1} \\ 1 & \vdots & \ddots & \vdots \\ 1 & x_{1n} & \cdots & x_{\text{ln}} \end{bmatrix} \qquad \mathbf{a}_i = [a_{i0}\ a_{i1} \cdots a_{il}]$$

For the local learning algorithm, the objective function is defined as a linear combination of squared error, which is a difference between output data and the result produced each fuzzy rule when considering the weighting factor matrix. $\mathbf{W}_i$ This matrix captures the activation levels of input data with respect to the i^{th} sub-space. In this sense, we can consider the weighting factor matrix as the discrete version of a fuzzy representation of the corresponding sub-space. The optimal coefficients of the consequent polynomial of i^{th} rule are described in a usual manner

$$\mathbf{a}_i = (\mathbf{X}_i^T \mathbf{W}_i \mathbf{X}_i)^{-1} \mathbf{X}_i \mathbf{W}_i \mathbf{Y} \tag{9}$$

Notice that the coefficients of the consequent polynomial of each rule have been computed independently using a subset of training data. Also, the computation can be

implemented in parallel meaning that the computing overhead becomes independent from the number of rules.

4 Optimization of the FCM-Based RBF Neural Network

The SEA is an evolutionary algorithm which evolutions subspace set in each generation [9-10]. The ideas for designing the algorithm come from evolutionary computation, but differ from the conventional evolutionary algorithm whose search method whose search methods handle some natural phenomena. It has been demonstrated that SEA is better than conventional evolutionary algorithms such as GA in the robustness of algorithms and the accuracy of solutions. SEA is a Branch-And-Bound Method based on subspace set. If the amount of subspaces divided by the whole solution space is large enough, the global optimization can be obtained by the algorithm. However, the amount of subspace sets is difficult to be huge quantities due to the limited time and constrained dimension. If a subspace set is replaced by one solution (point), then we have the SEA. The new subspace set is obtained by the evolutionary the M selected solutions (subspace sets).

The space evolutionary algorithm (SEA) is applied to parametric optimization such as the number of cluster, the fuzzification coefficient and Polynomial type.

5 Experimental Results

In all experiments, we randomly divide the data set into i) the training (60%) and validation (40%) and ii) training(50%), validation(30%) and testing(20%) part of data, respectively. To come up with a quantitative evaluation of the resulting neural network, we use the standard performance index of the Root Mean Square Error (RMSE) as expressed (10).

Table 1 includes a list of parameters used in the space evolutionary algorithm of the network used in the numeric experiments. Their numeric values were selected through a trial and error process by running a number of experiments and monitoring the pace of learning and assessing its convergence.

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(y_k - \hat{y}_k)^2} \tag{10}$$

Table 1. Parameters used in the space evolutionary algorithm of the nework

	MPG data	Boston housing data
Generation size	1000	
Population size	100	
M	8	
α	[-0.5 1.5]	
Number of all input variables	7	13

5.1 Automobile Miles Per Gallon (MPG)

We consider the well-known automobile MPG data (http://archive.ics.uci.edu/ml/datasets/auto+mpg) with the output being the automobile's fuel consumption expressed in miles per gallon. This dataset includes 392 input-output pairs (after removing incomplete data points). The number of input variables is 7 such as cylinder, displacement, horse power, weight, acceleration, model year, origin.

In the experiments for machine learning dataset, we make 10 experiments to get statistical result.

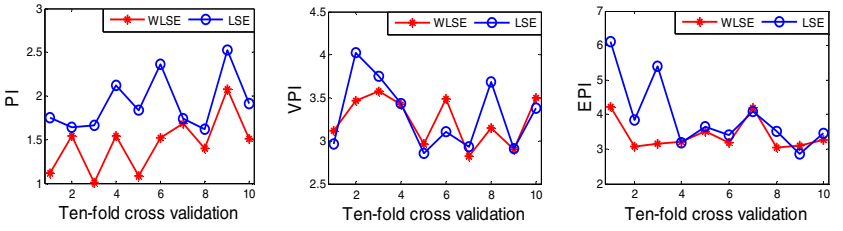


Fig. 2. Performance index of WLSE and LSE in the 10 experiments (3 split data set-training, validation, testing)

In Fig. 2 In case of three split dataset, the value of performance index of WLSE model related to both training data and testing data is much more superb than that of LSE model. The results show that the LSE model with optimization has much less over-fitting when compared with LSE model without optimization. In Comparison with LSE model as well as other models studied previously, WLSE model is preferable from the viewpoint of approximation as well as generalization capabilities.

5.2 Boston Housing Data

This data set concerns about real estate in the Boston area (ftp://ftp.ics.uci.edu/pub/machine-learnindatabases/housing/housing.data). The median value of the house (MEDV) is considered as an output variable. The input variables consist of 13 variables. In the experiments for machine learning dataset, we make 10 experiments to get statistical result.

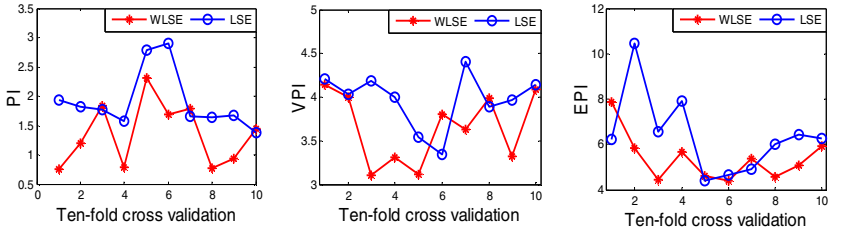


Fig. 3. Performance index of WLSE and LSE in the 10 experiments (3 split data set -training, validation, testing)

In Fig. 3(a), from the viewpoint of performance, WLSE model is much better than that of LSE model. In case of three split dataset, the value of performance index of WLSE model related to both training data and testing data is much more superb than that of LSE model. Table 2 shows that the LSE model with optimization has much less over-fitting when compared with LSE model without optimization. In Comparison with LSE model as well as other models studied previously, WLSE model is more preferable from the viewpoint of approximation as well as generalization capabilities.

Table 2. Comparative analysis of the performance of selected model

(AVG ± STD)

Model			Number of clusters		Performance index		
					PI	EPI_1(VPI)	EPI_2
RBFNN[11]			25		6.36 ± 0.24		6.94 ± 0.31
RBFNN with context-free clustering[11]			25		5.52 ± 0.25		6.91 ± 0.45
Linguistic Modeling [11]	Without Optimization		25		5.21 ± 0.12		6.14 ± 0.28
	One-loop Optimization		25		4.80 ± 0.52		5.22 ± 0.58
	Multi-step Optimization		25		4.12 ± 0.35		5.32 ± 0.96
Proposed model	Without optimization (m=2)	LSE	25	S	7.325 ± 0.285		7.331 ± 0.503
				L	0.490 ± 0.160		266.5 ± 236.4
				M	0.012 ± 0.040		142.0 ± 184.2
				Q	0.011 ± 0.035		58.93 ± 24.21
		WLSE	25	S	7.518 ± 0.288		7.465 ± 0.468
				L	2.186 ± 0.148		3.935 ± 0.466
				M	0.911 ± 0.069		6.503 ± 1.239
				Q	0.779 ± 0.039		7.647 ± 4.766
	With optimization	LSE	≤ 25	2s	1.938 ± 0.470	4.214 ± 0.498	
				3s	1.912 ± 0.513	3.971 ± 0.317	6.378 ± 1.775
		WLSE		2s	1.678 ± 0.615	3.688 ± 0.362	
				3s	1.347 ± 0.547	3.649 ± 0.408	5.378 ± 1.052

6 Concluding Remarks

In this paper, we have proposed optimization methodology of FCM-based RBFNN. The proposed model is the extended architecture of conventional RBFNN.

The performance of FCM-based RBFNN is affected by the type of polynomials as well as some parameters such as the number of cluster and fuzzification coefficient of FCM. The SEA is exploited to find the structural as well as parametric factors minimizing performance index of the proposed model.

Acknowledgements. This work was supported by National Research Foundation of Korea Grant funded by the Korean Government (NRF-2009-0074928) and also supported by the GRRC program of Gyeonggi province [GRRC SUWON2009-B2, Center for U-city Security & Surveillance Technology].

References

1. Bikdash, M.: A highly interpretable form of Sugeno inference systems. IEEE Trans. Fuzzy Syst. 7, 686–696 (1999)
2. Bezdek, J.C., Keller, J., Krisnapuram, R., Pal, N.R.: Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. Kluwer Academic Publisher, Dordrecht (1999)
3. Yager, R., Filev, D.P.: Generation of fuzzy rules by mountain clustering. Journal of Intelligent and Fuzzy Systems 2, 209–219 (1994)
4. Chiu, S.L.: Fuzzy model identification based on cluster estimation. J. Intel. and Fuzzy Syst. 2(3), 267–278 (1994)
5. Liu, F., Lu, P., Pei, R.: A new fuzzy modeling and identification based on fast-cluster and genetic algorithm. Intell. Control Autom. 1, 290–293 (2004)
6. Bastian, A.: Identifying fuzzy models utilizing genetic programming. Fuzzy Sets and Syst. 112, 333–350 (2000)
7. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Networks, vol. 4, pp. 1942–1948 (1995)
8. Roger Jang, J.S., Sun, C.T.: Functional equivalence between radial basis function networks and fuzzy inference systems. IEEE Trans. Neural Networks 4, 156–158 (1981)
9. Michalewiczz, S.: Evolutionary optimization for constrained parameter optimization problems. Evolutionary Computation 4, 1–32 (1996)
10. Michalewiczz, A.: Evoluationary optimization of contstrained problems. In: Sebald, A.V., Fogel, L.J. (eds.) Proc. of the 3rd Annual conf. on Evolutionary Programming. World Scientific, Singapore (1994)
11. Pedrycz, W., Kwak, K.C.: Linguistic models as a framework of user-centric system modeling. IEEE Trans. Syst. Man Cybern. A 36, 727–745 (2006)

Design of Information Granulation-Based Fuzzy Radial Basis Function Neural Networks Using NSGA-II

Jeoung-Nae Choi[1], Sung-Kwun Oh[2], and Hyun-Ki Kim[3]

[1] Department of Electrical Engineering, Daelim college, 526-7, Bisan-dong, Dongan-gu, Anyang-si, Gyeonggi-do, 431-717, South Korea
[2,3] Department of Electrical Engineering, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
ohsk@suwon.ac.kr

Abstract. This paper is concerned with information granulation-based fuzzy radial basis function neural networks (IG-FRBFNN) and its multi-objective optimization by means of the nondominated sorting genetic algorithms II (NSGA-II). By making use of the clustering results, the ordinary least square (OLS) learning is exploited to estimate the coefficients of polynomial. In fuzzy modeling, complexity and interpretability (or simplicity) as well as accuracy of model are essential issues. Since the performance of the IG-RBFNN model is affected by some parameters such as the fuzzification coefficient used in the FCM, the number of rules and the orders of polynomials of the consequent part of fuzzy rules, we require to carry out both structural as well as parametric optimization of the network. In this study, the NSGA-II is exploited to find the fuzzification coefficient, the number of fuzzy rules and the type of polynomial being used in each conclusion part of the fuzzy rules in order to minimize complexity and simplicity as well as accuracy of a model simultaneously.

Keywords: Fuzzy c-means clustering, nondominated sorting genetic algorithm II, fuzzy radial basis function neural network, ordinary least squares method.

1 Introduction

The problem of estimating an unknown function from sample data is one of the key issues in the field of fuzzy system modeling. The principal objective is to learn an unknown functional mapping between input and output data using a set of known training samples.

Fuzzy Radial Basis Function Neural Networks (FRBFNNs) are designed by integrating the principles of a Radial Basis Function Neural Network (RBFNN) and the Fuzzy C-Means (FCM) algorithm [1].

In this study, we considered multi-objective optimization of the IG-FRBFNN by means of the NSGA-II. There are some strictly different points with previous version of the optimization of the IG-FRBFNN [2].

The NSGA-II is exploited to find the fuzzification coefficient, the number of fuzzy rule and the types of polynomial being used in each conclusion part of the fuzzy rules minimizing complexity and simplicity as well as accuracy of a model simultaneously.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 215–222, 2010.

Section 2 describes an architecture and learning methods of the IG-FRBFNN. Section 3 deals with the NSGA-II and the multi-objective optimization of IG-FRBFNN using the NSGA-II. Section 4 presents the experimental results. Finally, some conclusions are drawn in Section 5.

2 Architecture and Learning of the IG-FRBFNN

The proposed IG-FRBFNN is the extended structure of the conventional FRBFNN and consists of the premise part and consequent part as shown in Fig. 1. We consider four types of polynomial such as constant type, linear type, quadratic type, and modified quadratic type as a local model representing input-output relationship in a subspace. One of four types is selected and used by means of optimization algorithm. Also, the extracted information granules by using FCM are used as the prototype of consequent polynomials of fuzzy rules to improve interpretability of local models. The advantages of an IG-RBFNN are that does not suffer from curse of dimensionality and provides a through the coverage of whole input space. In the sequel, we can construct more accurate model with a small number of fuzzy by using high-order polynomial. First, we recall Fuzzy C-Means Clustering to help comprehend the IG-FRBNN in the following session.

2.1 Fuzzy C-Means Clustering

The FCM clustering algorithm [13] is a set-partitioning method based on Picard iteration through necessary conditions for optimizing a weighted sum of squared errors objective function (J_m).

Fuzzy c-partitions of $\mathbf{X}$ satisfy the condition

$$\sum_{i=1}^{n} u_{ik} = 1, \qquad 1 \le k \le m \tag{1}$$

$$0 \langle \sum_{k=1}^{m} u_{ik} \langle m, \quad 1 \le i \le n \tag{2}$$

The FCM algorithm was developed to minimize the objective function

$$J_m = \sum_{i=1}^{n}\sum_{k=1}^{m} (u_{ik})^m d(\mathbf{x}_k, \mathbf{v}_i), \quad 1 \langle m \langle \infty \tag{3}$$

Where the membership functions $\{u_{ik}\}$ take values from the interval [0, 1] and satisfy the constraints (1) and (2). In (3), $d(\mathbf{x}_k, \mathbf{v}_i)$ is any inner product norm metric of the distance between the input vector $\mathbf{x}_k \in \mathbf{X}$ and the prototype (center value) $\mathbf{v}_i \in \Re^l$. The coupled first order necessary conditions for solutions $(\mathbf{U},\mathbf{V})$ of $min\{J_m(\mathbf{U},\mathbf{V})\}$ are

$$u_{ik} = w_{ik} = \frac{1}{\sum_{i=1}^{n} \left(\frac{\left\| \mathbf{x}_k - \mathbf{v}_i \right\|}{\left\| \mathbf{x}_k - \mathbf{v}_j \right\|} \right)^{(2/(m-1))}}, \quad 1 \le k \le m, \ 1 \le i \le n \tag{4}$$

and

$$v_i = \frac{\sum_{k=1}^{m} u_{ik}^m \mathbf{x}_k}{\sum_{k=1}^{m} u_{ik}^m}, \ 1 \le i \le n \tag{5}$$

2.2 Architecture of IG-FRBFNN

The IG-FRBFNN comes as an extended structure of the conventional FRBFNN and consists of the premise part and consequent part as shown in Fig. 1. The difference between the IG-FRBFNN and the conventional FRBFNN arises in consequent part. The extracted information granules (center values of each cluster) by using FCM are applied to consequent polynomials of fuzzy rules to improve interpretability of local models. The IG-RBFNN does not suffer from the curse of dimensionality (as all variables are considered *en block*). In the sequel, we construct more accurate model with a small number of fuzzy rule by using high-order polynomial.

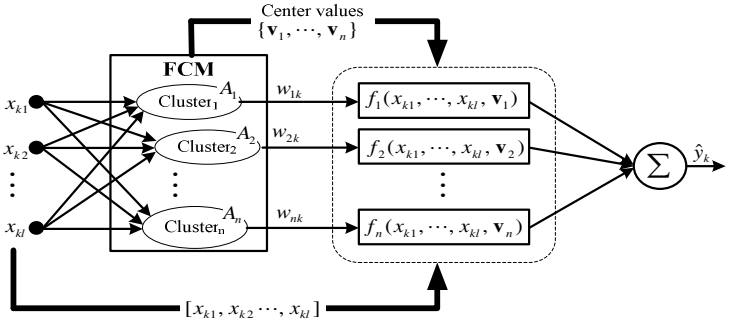


Fig. 1. Architecture of IG-FRBFNN

The IG-FRBFNN shown in Fig. 1 can be represented in form of "if-then" fuzzy rules

$$\mathbf{R}^i : IF \ \ \mathbf{x}_k \text{ is inclued in cluster } A_i \ THEN \ y_{ki} = f_i(\mathbf{x}_k, \mathbf{v}_i) \tag{6}$$

where, $\mathbf{R}^i$ is the *i*th fuzzy rule, *i*=1, ···, *n*, *n* is the number of fuzzy rules(the number of clusters), $f_i(\mathbf{x}_k, \mathbf{v}_i)$ is the consequent polynomial of the *i*th fuzzy rule, i.e., it is a local model representing input-output characteristic of the *i*th sub-space (local area). w_{ik} is the degree of membership (or activating level) of *i*th local model, that is calculated by FCM. $\mathbf{v}_i = \{v_{i1}, v_{i2}, \cdots, v_{il}\}$ is the *i*th prototype

The calculation of the numeric output of the model, based on the activation levels of the rules, uses the following expression

$$\hat{y}_k = \sum_{i=1}^{n} w_{ik} f_i(x_{k1}, \cdots, x_{kl}, \mathbf{v}_i) \tag{7}$$

2.3 Learning Algorithm of IG-FRBFNN

The premise and consequent part learning of the IG-FRBFNN is carried out sequentially by using the FCM and the ordinary least squares method (OLS) method.

OLS is to determine the coefficients of the model through the minimization of the objective function J_G. OLS is well known global learning algorithm that is oriented to minimize its overall squared error between real output and global model output.

$$J_G = \sum_{k=1}^{m} \left(y_k - \sum_{i=1}^{n} w_{ik} f_i(\mathbf{x}_k - \mathbf{v}_i) \right)^2 \tag{8}$$

where w_{ik} is the normalized firing strength (activation level) as expressed by (4).

The performance index J_G can be rearranged into a simple matrix form

$$J_G = (\mathbf{Y} - \mathbf{Xa})^T (\mathbf{Y} - \mathbf{Xa}) \tag{9}$$

Where $\mathbf{a}$ is the vector of coefficients of consequent polynomial, $\mathbf{Y}$ is the output vector of real data, $\mathbf{X}$ is matrix which is rearrange with input data, information granules(centers of each cluster) and activation level.

The coefficients of the consequent polynomial of fuzzy rules can be determined in a usual manner that is

$$\mathbf{a} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y} \tag{10}$$

3 Multiobjective Optimization of the IG-FRBFNN

The objective of effective learning method is to create almost ideal IG-FRBFNN satisfying accuracy as well as simplicity and complexity. Here we consider the structural as well parametric optimization realized by means of the NSGA-II. We start with the introduction of the NSGA-II and then discuss the arrangement and interpretation of an individual and Multiobjective functions for the optimization of the IG-FRBFNN.

3.1 Nondominated Sorting Genetic Algorithm - II

Multi-objective genetic algorithms, called NSGA-II (non-dominated sorting genetic algorithm II) recently proposed by Deb, Pratap, Agarwal, and Meyarivan [12]. The general principle of NSGA-II: at each generation t, a parent population P_t of size N and an offspring population Q_t of the same size are merged for forming a population R_t (R_t=$P_t \cup Q_t$) of size 2N. Then, the population R_t is partitioned into a number of sets called fronts F, which are constructed iteratively. Front F_1 consists of the non-dominated solutions from R_t . F_2 consists of non-dominated solution from the set ($R_t - F_1$) and so on. In general, F_i consists of the non-dominated solutions from the set (R_t –($F_1 \cup F_2 \ldots \cup F_{i-1}$)).

Deb et al. have proposed a fast partitioned algorithm called fast non-dominated sorting algorithm [12]. Once all fronts are identified, a new parent population P_{t+1} of size N is formed by adding the fronts to P_{t+1} in order (front one F_1 followed by front two and so on) as long as the size of R_t do not exceed N individuals. If the number of individuals present in P_{t+1} is lower than N, a crowding procedure is applied to the first

front F_i not included in P_{t+1}. The aim of this procedure is to insert the N - P_{t+1} best individuals which miss to fill all population R_t . For each solution i in F_i a crowding distance d_i is calculated based on each objective Ob_k. The front F_i is sorted according to objective function value Ob_k in ascending order of magnitude. The first and the last individuals of the front are assigned infinity as their distance with respect to Ob_k. For all other intermediate individuals are assigned a distance value equal to the absolute normalized difference in the objective function values of two adjacent solutions. This calculation is repeated for other objective functions. The overall crowding distance for each solution i is calculated as the sum of individual distance values corresponding to each objective function. The individuals with the highest crowding distance values are added to P_{t+1} until $P_{t+1} = N$. Once the individuals appertaining to the population P_{t+1} are identified, a new population Q_{t+1} of size N is created by selection, crossover and mutation using individuals of the population P_{t+1}. It is important to note that with NSGA-II, the selection procedure is based on a crowded-comparison operator defined as follows: to select an individual, two solutions are chosen randomly and uniformly from P_{t+1} and that of smaller number of front i_{rank} is preserved. If both solutions belong to the same front, one separates them by calculating the crowding distance for each solution and the solution with the higher distance is preferred. The NSGAII operation will repeat the procedure until the stopping criteria is satisfied.

3.2 Arrangement and Interpretation of Individual in the NSGA-II

In the NSGA-II, an individual is represented as a vector involving the fuzzification coefficient (①), the number of fuzzy rule(②) and the type of polynomial of consequent part for each fuzzy rule(③) as illustrated in Fig. 2(a). The size of ③ is the same with the upper boundary of search space of the number of fuzzy rule.

Fig. 2(b) offers an interpretation of the content of the particle in case the upper boundary of search space of the fuzzy rule is 8. As far as interpretation is concerned, the number of fuzzy rules, and the orders of polynomials have to be integer number but these values are real number. So we round off these values to the nearest integer. The fuzzification coefficient is 1.1, the number of fuzzy rule is 6 and the first six values of part ③ are selected. So, the form of first local model is linear type and the forms of five local models are quadratic type.

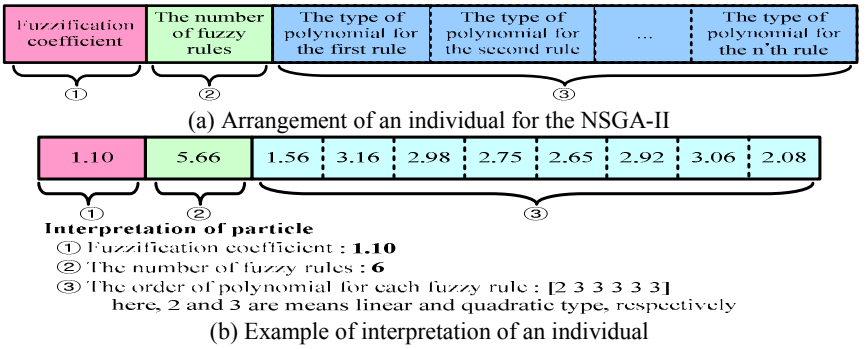


Fig. 2. Individual composition of NSGA-II and interpretation

3.3 Objective Functions of IG-FRBFNN

Three objective functions are used to evaluate the accuracy, the complexity and the interpretability of an IG-FRBFNN. The three objective functions are the root mean squared error (RMSE), entropy of partition and the total number of coefficients of polynomials to be estimated.

The RMSE is the accuracy criterion of the IG-RBFNN. The RMSE is given as

$$E = \sqrt{\frac{1}{m}\sum_{k=1}^{m}(y_k - \hat{y}_k)^2} \tag{11}$$

As a measure for evaluating the structure complexity of a model we consider the entropy of the partition [14]. The entropy of partition is a degree of overlapping between the fuzzy regions and it can be calculated using a degree of membership. Considering all samples of the training dataset, the entropy of a partition is a given by

$$H = -\sum_{j=1}^{n}\sum_{k=1}^{m} w_{ik}\log(w_{ik}) \tag{12}$$

As a simplicity criterion we consider the total number of coefficients of local models.

In a nutshell, we find the Pareto optimal sets and Pareto front minimizing {E, H, N} by means of the NSGA-II. This leads to easily interpretable, simple and accurate fuzzy model.

4 Experimental Study

The IG-FRBFNN is applied to nonlinear function approximation problems. We consider the following two 2-D nonlinear functions that is

Example 1

$$y(x_1, x_2) = 1.9\left(1.35e^{x_2}e^{-x_1}\sin(12(x_2 - 0.6)^2)\sin(7x_1)\right), \quad 0 \le x_1, x_2 \le 1 \tag{13}$$

Example 2

$$y(x_1, x_2) = \left(x_1^{1.5} - 1.5\sin(3x_2)\right)^2, \quad 0 \le x_1, x_2 \le 3 \tag{14}$$

We considered 400 input–output data pairs that are extracted randomly for each example. The construction of the fuzzy model is completed for 200 data points being regarded as a training set. The rest of the data set (i.e., 200 data points) is retained for testing purposes.

The NSGA-II just provides the Pareto optimal sets, which are non-dominated solutions. Selecting the optimal solution within the Pareto optimal sets is different from the NSGA-II. If we place more emphasis on the accuracy than the simplicity, then we define the individual having the best accuracy within Pareto optimal set to be the best solution. Table 1 summarizes the performance values of individual that has the best accuracy within Pareto optimal set.

Table 1. The summary of results of the optimized IG-FRBFNN

	No. of rules	Orders of each polynomial *	Fuzzification coefficient	Objective values for training data			Objective values for testing data		
				E	H	N	E	H	N
Ex 1	7	[4 3 3 3 3 4 3]	1.97	0.10	234.14	38	0.13	93.34	12
Ex 2	7	[3 2 2 3 3 3 2]	1.71	0.458	99.49	33	1.15	110.87	33

*) 1: Constant form, 2: Linear form, 3: Quadratic form, 4: Modified quadratic form.

5 Conclusions

In this study, we have introduced a new class of models, IG-FRBFNN and proposed the design methodology to generate the accurate, simple and easily interpretable IG-FRBFNN by means of the NSGA-II. The IG-FRBFNN is the extended architecture of the conventional FRBFNN. The performance of IG-FRBFNN is affected by some parameters such as the values of the fuzzification coefficient of the FCM, the number of rules and the order of polynomial in the consequent parts of rules. The optimization of the IG-FRBFNN is focused on multi objective such as the complexity and the simplicity and the accuracy, The NSGA-II was exploited as a multi-objective optimization vehicle to carry out the structural as well as parametric optimization of the IG-FRBFNN. The effectiveness of the IG-FRBFNN have been investigated and analyzed through two examples for function approximation.

Acknowledgements. This work was supported by National Research Foundation of Korea Grant funded by the Korean Government (NRF-2009-0074928) and also supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded by Korea Small and Medium Business Administration in 2009.

References

1. Mitra, S., Basak, J.: FRBF: A Fuzzy Radial Basis Function Network. Neural Comput. Applic. 10, 244–252 (2001)
2. Choi, J.N., Lee, Y.I., Oh, S.K.: Fuzzy Radial Basis Function Neural Networks with Information Granulation and Its Genetic Optimization. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5552, pp. 127–134. Springer, Heidelberg (2009)
3. Oh, S.K., Lee, I.T., Choi, J.N.: Design of fuzzy polynomial neural networks with the aid of genetic fuzzy granulation and its application to multi-variable process system. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 774–779. Springer, Heidelberg (2006)
4. Choi, J.N., Oh, S.K., Pedrycz, W.: Structural and parametric design of fuzzy inference systems using hierarchical fair competition-based parallel genetic algorithms and information granulation. Int. J. Approx. Reaso. 49, 631–648 (2008)
5. Lin, F.J., Teng, L.T., Lin, J.W., Chen, S.Y.: Recurrent Functional-Link-Based Fuzzy-Neural-Network-Controlled Induction-Generator System Using Improved Particle Swarm Optimization. IEEE Trans. Indust. Elect. 56(5), 1557–1577 (2009)
6. Bastian, A.: Identifying fuzzy models utilizing genetic programming. Fuzzy Sets and Syst. 112, 333–350 (2000)
7. Jin, Y.: Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement. IEEE Trans. Fuzzy Syst. 8(2), 212–221 (2000)

8. Setnes, M., Roubos, H.: GA-based modeling and classification: complexity and performance. IEEE Trans. Fuzzy Syst. 8(5), 509–522 (2000)
9. Guenounou, O., Belmehdi, A., Dahhou, B.: Multi-objective optimization of TSK fuzzy models. Expert systems with applications 36, 7416–7423 (2009)
10. Knowles, J., Corne, D.: Approximating the nondominated front using the Pareto archived evolution strategy. Evol. Computing 8, 149–172 (2000)
11. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms-A comparative case study. In: Parallel Problem Solving from Nature, pp. 292–301 (1998)
12. Deb, K., Agrawal, S., Pratab, A., Meyarivan, T.: A fast elitist nondominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Proc. Parallel Problem Solving from Nature VI Conference, pp. 849–858 (2000)
13. Bezdek, J.C., Keller, J., Krisnapuram, R., Pal, N.R.: Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. Kluwer Academic Publisher, Dordrecht (1999)
14. Delgado, M., Ceullar, M.P., Pegalajar, M.C.: Multiobjective Hybrid Optimization and Training of Recurrent Neural Networks. IEEE Trans. Syst. Man cybern. –Part B 38(2), 381–403 (2008)

Practical Criss-Cross Method for Linear Programming

Wei Li*

Institute of Operational Research & Cybernetics, Hangzhou Dianzi University, Hangzhou, 310018, P.R. China

Abstract. In this paper we first generalize the concept of pivoting index, proposed by Pan, to the standard form of linear programming problems by considering its dual problem. Then, we develop a practical variant of criss-cross pivot algorithm for linear programming problem. The new criss-cross algorithm is not only finiteness, but also more efficient in practice.

Keywords: Linear programming; Criss-cross pivot method; Pivoting index.

1 Introduction

Simplex pivot methods always require and preserve either primal or dual feasibility of the generated basic solutions. To produce a primal or dual feasible basic solution is a nontrivial task. It require the solution of another linear programming problem, the so-called phase-I problem. The criss-cross method was first introduced by Zionts [1] in 1969 as a pivot algorithm for solving linear programming requiring no feasibility of the basis. It can be initialized by an arbitrary basic solution. The first finite criss-cross algorithm was independently proposed by Chang, Terlaky and Wang [2,3]. The main advantage of the criss-cross method is its simplicity. It can be started with any basic solution and solves the linear programming in one phase, in a finite number of pivot steps. However, in spite of their elegance and simplicity, to date criss-cross methods are not efficient in practice. [4].

In this paper we introduce a practical variant of the finite criss-cross method. By technique of reordering variable indices, this new criss-cross method is more efficient in practice. The finiteness of the new criss-cross pivot variant is proven. The rest of the paper is organized as follows. In Section 2, we generalize the concept of pivoting index, proposed by Pan[5], to the standard form of linear programming problems from its dual side.Then a new criss-cross method is established with the proof of its finiteness. In Section 3 we report our computational results with two group of linear programming problems, and Section 4 concludes the paper.

* This work was partially supported by NSF of China (Mathematic Tian Yuan fund) 10926058.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 223–229, 2010.

2 Least-Pivoting-Index Criss-Cross Method

Consider the following standard linear programming problem:

$$min\ c^T x \tag{1}$$

$$s.t.\ Ax = b,$$

$$x \geq 0$$

and its associated dual problem

$$max\ \ b^T y \tag{2}$$

$$s.t.\ A^T y + s = c, \quad s \geq 0,$$

where $c, x, s \in R^n$, $b, y \in R^m$, $A \in R^{m \times n}$ with $m < n$, $rank(A) = m$. Denote

$$A = [a_1, \ldots, a_n]$$

A basis J_B is an index set with the property that $J_B \subseteq \{1, \ldots, n\}, \mid J_B \mid = m$ and $A_B := [a_i \mid i \in J_B]$ is an invertible matrix. Nonbasis J_N is the complement of J_B, i.e. $J_N := \{1, \ldots, n\} \backslash J_B$. Further, let A_N denote the complement matrix, the nonbasic part of the matrix A. Given a basis J_B, we call the following matrix a simplex tableau with respect to J_B:

-z	$\ldots$	s_j	$\ldots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_j	$\ldots$	$\bar{a}_{ij}$	$\ldots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

where $[\bar{a}_{ij}]_{\mid J_B \mid \times \mid J_N \mid} = A_B^{-1} A_N,\ [s_j | j \in J_N] = c_N^T - c_B^T A_B^{-1} A_N,\ [x_i | i \in J_B]^T = A_B^{-1} b,\ z = c_B^T A_B^{-1} b.$

To indicate the relation with the basis, in the later part of the paper we will denote x by $x(B)$, and s by $s(B)$. Furthermore, we extend $x(B)$ and $s(B)$ to their full dimensions, i.e., we let

$$x(B)_i := \begin{cases} x(B)_i & \text{for } i \in J_B, \\ 0 & \text{for } i \in J_N, \end{cases} \quad s(B)_i := \begin{cases} s(B)_i & \text{for } i \in J_N, \\ 0 & \text{for } i \in J_B. \end{cases}$$

The first finite criss-cross algorithm proceeds as follows:

Algorithm 1 (Least index criss-cross algorithm)

0. Get a basis J_B to start with.

1. Let $I = \{i | x(B)_i < 0\}$ and $J = \{i | s(B)_i < 0\}$. If $I \bigcup J = \emptyset$, then J_B is optimal, stop.

2. Let $k = min\{k | k \in I \bigcup J\}$. If $k \in I$,then go to 3; If $k \in J$, then go to 4.

3. Let $S = \{j | j \in J_N |$ and $\bar{a}_{kj} < 0\}$. If $S = \emptyset$, then the primal problem (1) has no feasible solution, stop. Otherwise, let $j = min\{j | j \in S\}$. Let $J_B := J_B \bigcup \{j\} \backslash \{k\}$ and go back to 1.

4. Let $T = \{i | i \in J_B |$ and $\bar{a}_{ik} > 0\}$. If $T = \emptyset$, then the dual problem (2) has no feasible solution, stop. Otherwise, let $i = min\{i | i \in T\}$. Let $J_B := J_B \bigcup \{k\} \backslash \{i\}$ and go back to 1.

The following result was shown in [2,3].

Theorem 1. Algorithm 1 will terminate in a finite number of iterations.

In spite of their elegance and simplicity, unfortunately, just as T. Illés et al wrote in [4]: "to date criss-cross methods are not efficient in practice." Now a simple fundamental question arises: Is there any practical finite criss-cross method? Motivated by seeking a effective criss-cross method, we have the following consideration.

Among a number of candidates to enter or leave the basis, an "ideal" pivoting rule, if exists, should be able to recognize and take an optimal basic variable to enter and/or an optimal non-basic variable (non-basic at the optimal solution) to leave the basis. To this end, the pivoting index was first introduced by Pan [5] in 1990 for the linear programming problem:

$$\begin{aligned} & max\ c^T x \\ & s.t.\ a_i^T x \geq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \qquad j = 1, \ldots, n, \end{aligned} \tag{3}$$

where $c = (c_1, \ldots, c_n)$ and $a_i,\ i = 1, \ldots, m$ are nonzero column vectors.

Definition 1. The pivoting index α_j of variable x_j is defined by

$$\alpha_j = c_j, \quad j = 1, \ldots, n, \tag{4}$$

$$\alpha_j = (ca_{j-n}) / \mid a_{j-n} \mid, \quad j = n+1, \ldots, n+m, \tag{5}$$

where $\mid \cdot \mid$ denotes the 2-norm, and $x_{n+1}, \ldots, x_{n+m}$ are slack variables corresponding to the constraints $a_i^T x \geq b_i, \quad i = 1, \ldots, m$, respectively.

It is noticeable that, pivoting indices have full geometric meaning. In fact, the quantity $\alpha_j / \mid c \mid$, for each $j = 1, \ldots, n+m$, is equal to the cosine of the angle between two gradients of the objective function and the left side function of the "j-t" inequality constraint, i.e.,

$$x_j \geq 0,\ for \quad j = 1, \ldots, n, or \tag{6}$$

$$a_{j-n}^T x_j \geq b_{j-n}, \quad for \quad j = j+1, \ldots, n+m, \tag{7}$$

This as well gives of correspondence between α_j and x_j and the j-th constraint, and hence α_j is also referred to as the pivoting index of the j-th constraint.

The following result was first shown in [5] and developed in [6,7].

Statement 1. Optimal basic variables of problem 3 tend to be characterized by bearing higher pivoting indices, and optimal nonbasic variables by bearing lower pivoting indices.

This crash heuristic method is very effective, it clearly can create a *good* initial basis in general, and can even create an optimal basis for linear programming on some occasions [5,6,7].

The concept of pivoting index, though defined for problem (3), can be easily generalized to the stand form of the linear programming (1) by considering its dual problem.

Definition 2. The pivoting index α_j of variable x_j is defined by

$$\alpha_j = (b^T a_j)/ \mid a_j \mid, \quad j = 1, \ldots, n. \tag{8}$$

By utilizing Statement 1 and the theory of complementary slackness, we obtain

Statement 2. Optimal basic variables of problem (1) tend to be characterized by bearing lower pivoting indices, and optimal nonbasic variables by bearing higher pivoting indices.

Now we are ready to present our new variant of finite criss-cross algorithm, incorporating the least-pivoting-index rule instead of least index rule.

Algorithm 2 (Least pivoting index criss-cross algorithm)

0. Get a basis J_B to start with.

1. Let $I = \{i | x(B)_i < 0\}$ and $J = \{i | s(B)_i < 0\}$. If $I \bigcup J = \emptyset$, then J_B is optimal, stop.

2. Let $k = Argmin\{\alpha_k | k \in I \bigcup J\}$. In case of a tie, we simply break this tie by selecting one having the smallest ordinary index. If $k \in I$,then go to 3; If $k \in J$, then go to 4.

3. Let $S = \{j | j \in J_N |$ and $\bar{a}_{kj} < 0\}$. If $S = \emptyset$, then the primal problem (1) has no feasible solution, stop. Otherwise, let $j = Argmin\{\alpha_j | j \in S\}$. In case of a tie, break this tie by selecting one having the smallest ordinary index. Let $J_B := J_B \bigcup \{j\} \backslash \{k\}$ and go back to 1.

4. Let $T = \{i | i \in J_B |$ and $\bar{a}_{ik} > 0\}$. If $T = \emptyset$, then the dual problem (2) has no feasible solution, stop. Otherwise, let $i = Argmin\{\alpha_i | i \in T\}$. In case of a tie, break this tie by selecting one having the smallest ordinary index. Let $J_B := J_B \bigcup \{k\} \backslash \{i\}$ and go back to 1.

Theorem 2. Algorithm 2 will terminate in a finite number of iterations.

Proof. Reindex the variables in problem (1) from 1 up to n, according to mono-increase order of pivoting indices; and in case when there are several variables with equal pivoting indices, reindex them according to their index mono-increase order. Then, the algorithm 1 and algorithm 2 are equivalent for the problem, implying finiteness of algorithm 2 since algorithm 1 is finite.

3 Computational Results

Computational tests have been performed to gain an insight into the numerical behavior of the proposed method. Algorithm 2 was tested on two groups of linear programming problems and compare with Algorithm 1. The first group includes 60 randomly constructed problems, with size $m + n$ from 7 to 49. The second group involves 5 sparse NETLIB problems that do not have BOUNDS and RANGES sections in their MPS files, since the current version of our codes cannot handle such problems implicitly. In our tests, the following 2 codes in C++ 6.0 were tested, and compared with each other:

C1. Algorithm 1;
C2. Algorithm 2.

The machine precision was about 16 decimal places. For each code, tolerance used was 10^{-6}. We first report our test result of a small size problem *in detail* solved by code C1 and code C2 respectively with the same initial basis.

Example 1

$$\begin{aligned} max\ z = 3x_1 + 5x_2 - x_3 - x_4, \\ x_1 + x_2 + x_3 + x_4 \leq 40, \\ 5x_1 + x_2 \leq 12, \\ x_3 + x_4 \geq 5, \\ x_3 + 5x_4 \leq 50, \\ x_i \geq 0, i = 1, \ldots, 3. \end{aligned}$$

Introducing slack variable and performing Gauss-Jordan pivoting operations, it is easy to obtain the initial tableau as follows:

Initial tableau

191.500	0.000	0.000	0.000	0.000	5.500	-0.500	4.500	0.000
-5.750	1.000	0.000	0.000	0.000	-0.250	0.250	-0.250	0.000
40.750	0.000	1.000	0.000	0.000	1.250	-0.250	1.250	0.000
-6.250	0.000	0.000	1.000	0.000	0.000	0.000	-1.250	-0.250
11.250	0.000	0.000	0.000	1.000	0.000	0.000	0.250	0.250

The interested reader can verify that from the initial basis (x_1, x_2, x_3, x_4) a solution process by algorithm 2 require only 3 iterations, consisting of the following sequence of basic solutions: $(x_5, x_2, x_3, x_4), (x_5, x_2, x_7, x_4), (x_3, x_2, x_7, x_4)$, whereas the solution process from the same initial basis by algorithm 1 consists of 6 iterations: (x_5, x_2, x_3, x_4), (x_5, x_2, x_7, x_4), (x_5, x_2, x_7, x_3), (x_1, x_2, x_7, x_3), (x_1, x_4, x_7, x_3), (x_2, x_4, x_7, x_3). Both algorithms reach the same optimal solution with the optimal value -88.

Numerical results for all problems are summarized in Table 1.

Table 1.

Problems	iteration	
	C1	C2
Total for Group 1	653	596
AFIRO: m=28 n=32	65	38
SC50B: m=51 n=48	61	61
SC50A: m=51 n=48	55	56
SC105: m=106 n=103	137	118
SC205: m=206 n=203	267	243
Total for Group 2	585	516

From Table 1 it is seen that C2 outperformed C1 with the iteration ratio 516/585. Also, it is seen that the algorithm C2' superiority over C1 grows with increase of problem sizes, overall.

4 Conclusions

The inefficiency of least-index criss-cross method arises from the fact that the basic variables will tend to become lower-indexed, a tendency which does not coincide with the real case in general since we can say nothing about indices of optimal basic variables (ones which are basic at the optimal solution) of a general problem.

Contrary to least-index criss-cross method, the new finite criss- cross method enjoys full of geometrical meaning in selecting pivot element. The priority of being chosen to enter the basis has been given to variables with the smaller pivoting-index, which is favorable from the spirit of the geometric characterization of an optimal basis. Moreover, it has the same advantage of simplicity, and at the same time, is free of the drawback of index-dependency. Consequently, it is more efficient. Table 1 shows that our theoretical analysis coincide with the numerical test results. The performance of algorithm 2 appears to be better than that of algorithm 1.

Certainly, drawing general and final conclusions require a more extensive computational study with large-scale problems, as is only possible with some sparsity-exploiting version of the new method. At least, however, we feel safe to conclude that to the proposed algorithm is attractive, and deserves a further investigation.

References

1. Zionts, S.: The criss-cross method for solving linear programming problems. Management Science 15, 426–445 (1969)
2. Terlaky, T.: A convergent criss-cross method. Mathematics of operationsforschung und statistik series Optimazation 16, 683–690 (1985)

3. Wang, Z.: A conformal elimination-free algorithm for oriented matroid programming. Annals of Mathematics 8, 51–59 (1987)
4. Illés, T., Terlaky, T.: Pivot versus interior point methods: Pros and cons. European Journal of Operational Research 140, 170–190 (2002)
5. Pan, P.Q.: Practical finite pivoting rules for the simplex method. OR Spektrum 12, 219–225 (1990)
6. Pan, P.Q., Li, W., Wang, Y.: A Phase-I Algorithm Using the Most-Obtuse-Angle Rule for the Basis-Deficiency-Allowing Dual Simplex Method. OR Transaction 8, 88–96 (2004)
7. Li, W.: A Note on Two Direct Methods in Linear Programming. European Journal of Operational Research 158, 262–265 (2004)

Calculating the Shortest Paths by Matrix Approach*

Huilin Yuan[1] and Dingwei Wang[2]

[1] Department of Commerce and Trade of Northeast University at Qinhuangdao
066004 Qinhuangdao, China
yhl@mail.neuq.edu.cn
[2] Information Department of Northeast University
110006 shenyang, China
dwwang@mail.neu.edu.cn

Abstract. The paper present a new approach of calculating the shortest paths, the approach was named as SPM (the shortest paths' matrix). Compared to the past approaches, SPM introduced distances into a matrix and could show activity distances directly in elements of the matrix and improved efficiency. The definition, operating rules of SPM and others related to SPM were described in this paper. At the end, the approach was applied to practical processes, the shortest paths were found out quickly. The approach provided a new resolution to the calculation of the shortest paths.

Keywords: BPR, Optimization, Shortest path, Reachable matrix, Activity distance.

1 Introduction

There are several traditional methods of calculating paths; they are 'Dijkstra', 'Bellman-Ford' and 'Floyd'. The 'Dijkstra' approach is suitable for calculating the shortest paths (SP) in a graph which has a single source and all the edge weight of the graph are positive; if one edge weight is negative, the result will be wrong by the approach. The 'Bellman-Ford' approach stores the shortest paths in a vector every time. The similarity between the above two approaches is that they only show paths in a graph which has a single source, so they can't be used widely.

In the 'Floyd' approach, there is a matrix $\boldsymbol{A}^k$ which stores the shortest paths calculated k-th times, so it solves the problem that the above two only calculate the paths under a single source. But the matrix in this approach must be calculated n times(if there is n points in a graph), n^2 elements must be involved every time, and n^3 steps are needed, efficiency of the approach is low[1-3], so the approach can not be fit for the business process reengineering(BPR) [4-5].

Under the situation, a new approach-the shortest path matrix (SPM) is proposed in the paper. It not only can get the short distances between n points, but also improve

* This work is partially supported by The National Natural Science Fund Grant #70771021 and Grant #60821063 to Prof. Dingwei Wang.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 230–237, 2010.

the efficiency compared to the above three approaches. The following are descriptions and rules relative to the SPM:

2 SPM

The SPM gains enlightenment from the reachable matrix. The reachable matrix is calculated on adjacent matrix $\boldsymbol{A}$ whose element is 0 or 1 and unit matrix $\boldsymbol{I}$ [6]. The elements a_{ij} and a_{jk} on ($\boldsymbol{I}$+$\boldsymbol{A}$) are 1, that means there is a path from i to k through j, and the element $a^2{}_{ik}$ ($a^2{}_{ik} = \sum_{j=1}^{n} (a_{ij} \oplus a_{jk})$) on the result matrix ($\boldsymbol{I}$+$\boldsymbol{A}$)^2 is 1, and ($\boldsymbol{I}$+$\boldsymbol{A}$)^2 describes the reach-ability between i to k through one intermediate point (or two paths) [7]. If the elements on $\boldsymbol{A}$ are replaced by distances, the elements on the result matrix must correspondingly show distances. Some concerned rules and matrices in the new concept are deferent with before and described as follows:

2.1 The Relative Conception

1) Network adjacent matrix $\boldsymbol{A}$: $\boldsymbol{A}$ is gained as following equation:

$$A = [a_{ij}]_{n \times n}, a_{ij} = \begin{cases} W(i,j), & \text{if } i \neq j \text{ and } i \text{ is adjacent to j;} \\ 0, & \text{if } i = j; \\ \infty, & \text{else.} \end{cases} \tag{1}$$

W_{ij} is weight of an edge.

2) Multiply-add operation: If $0 < a_{ij}, a_{jk} \ll \infty$, there is a path from i to k through j and the distance between i and k is $a_{ij} + a_{jk}$. And then the minus value in all distances from i to k through different j are gained. In a similar way, the minus distances between all points are gained by changing i and k.

The operation is similar to matrix multiplication (the row vector adds the corresponding column one) and is called multiply-add operation (denoted by $\times_{(+)}$). Matrix $\boldsymbol{A}$ multiply-add matrix $\boldsymbol{B}$ is matrix $\boldsymbol{C}$, the elements on $\boldsymbol{C}$ are calculated according to the following equation: $a_{ij} + a_{jk}$:

$$c_{ik} = \min_{j=0}^{n-1} (a_{ij} + b_{jk}) \tag{2}$$

3) r-th degree Multiply-added matrix: If the above matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are the same, the matrix $\boldsymbol{C}$ is called second multiply-added matrix (denoted by $\boldsymbol{A}^{2+}$). It describes the reach-ability and show the distances between points through no more than one intermediate point (or pl (path length) ≤2). In a similar way, the r-th degree multiply-added matrix $\boldsymbol{A}^{r+}$ is gained:

$$A^{r+} = A^{(r-1)+} \times_{(+)} A \tag{3}$$

The element $a^{r+}{}_{ik}$ on the matrix describes the reach-ability and shows the minus distance between points when $pl \leq r$.

4) SPM: Through calculation as showed above, when $\boldsymbol{A} \neq \boldsymbol{A}^{2+} \neq \boldsymbol{A}^{3+} \ldots \neq \boldsymbol{A}^{(r-1)+} = \boldsymbol{A}^{r+}$, we gain SPM ($R_p$):

$$R_p = A^{(r-1)+} \quad (4) \tag{4}$$

It shows the minus distances between all points.

2.2 The Calculating Process

Step.1: Fill in the vacancy of adjacent matrix $\boldsymbol{A}$ with '0' and '1'. Put '1' on a_{ij} (the intersection of two activities) which have information flow from activity i to j and put '0' on which have no information flow.

Step.2: If all the elements of a column on $\boldsymbol{A}$ are zero, the activity responding to this column should be executed as early as possible, because it is the original point of the whole activity process and does not need any information of other activities. We adjust it on the first column and row on $\boldsymbol{A}$.

If all the elements of a row are zero, the activity responding to this row should be executed behind other activities, because it is the terminal point and provides no information to the others. We put it on the last column and row on $\boldsymbol{A}$[8] [9-10].

In a similar way, we can adjust other activities on $\boldsymbol{A}$ in turn.

Step.3: $\boldsymbol{A}$ is translated into a network adjacent matrix according to equation (1).

Step.4: $\boldsymbol{A}^{r+}$ and R_p are gained according equations (2), (3) and (4).

2.3 Calculating SP of a Diagraph

The matrices of Fig.1 are gained according to Section 2.2.

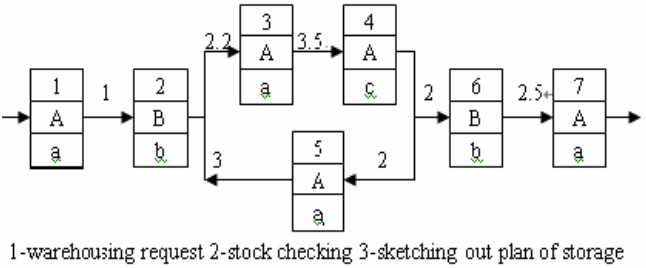


Fig. 1. Simplified storage process

$$A=\begin{pmatrix}0&1&\infty&\infty&\infty&\infty&\infty\\ \infty&0&2.2&\infty&\infty&\infty&\infty\\ \infty&\infty&0&3.5&\infty&\infty&\infty\\ \infty&\infty&\infty&0&2&2&\infty\\ \infty&\infty&3&\infty&0&\infty&\infty\\ \infty&\infty&\infty&\infty&\infty&0&2.5\\ \infty&\infty&\infty&\infty&\infty&\infty&0\end{pmatrix}\quad A^{2+}=\begin{pmatrix}0&1&3.2&\infty&\infty&\infty&\infty\\ \infty&0&2.2&5.7&\infty&\infty&\infty\\ \infty&\infty&0&3.5&5.5&5.5&\infty\\ \infty&\infty&5&0&2&2&4.5\\ \infty&\infty&3&6.5&0&\infty&\infty\\ \infty&\infty&\infty&\infty&\infty&0&2.5\\ \infty&\infty&\infty&\infty&\infty&\infty&0\end{pmatrix}$$

$$A^{3+}=\begin{pmatrix}0&1&3.2&6.7&\infty&\infty&\infty\\ \infty&0&2.2&5.7&7.7&7.7&\infty\\ \infty&\infty&0&3.5&5.5&5.5&8\\ \infty&\infty&5&0&2&2&4.5\\ \infty&\infty&3&6.5&0&8.5&\infty\\ \infty&\infty&\infty&\infty&\infty&0&2.5\\ \infty&\infty&\infty&\infty&\infty&\infty&0\end{pmatrix}\quad A^{4+}=\begin{pmatrix}0&1&3.2&6.7&8.7&8.7&\infty\\ \infty&0&2.2&5.7&7.7&7.7&10.2\\ \infty&\infty&0&3.5&5.5&5.5&8\\ \infty&\infty&5&0&2&2&4.5\\ \infty&\infty&3&6.5&0&8.5&11\\ \infty&\infty&\infty&\infty&\infty&0&2.5\\ \infty&\infty&\infty&\infty&\infty&\infty&0\end{pmatrix}$$

$$A^{5+}=\begin{pmatrix}0&1&3.2&6.7&8.7&8.7&11.2\\ \infty&0&2.2&5.7&7.7&7.7&10.2\\ \infty&\infty&0&3.5&5.5&5.5&8\\ \infty&\infty&5&0&2&2&4.5\\ \infty&\infty&3&6.5&0&8.5&11\\ \infty&\infty&\infty&\infty&\infty&0&2.5\\ \infty&\infty&\infty&\infty&\infty&\infty&0\end{pmatrix}\quad A^{6+}=\begin{pmatrix}0&1&3.2&6.7&8.7&8.7&11.2\\ \infty&0&2.2&5.7&7.7&7.7&10.2\\ \infty&\infty&0&3.5&5.5&5.5&8\\ \infty&\infty&5&0&2&2&4.5\\ \infty&\infty&3&6.5&0&8.5&11\\ \infty&\infty&\infty&\infty&\infty&0&2.5\\ \infty&\infty&\infty&\infty&\infty&\infty&0\end{pmatrix}$$

Because $A\neq A^{2+}\neq A^{3+}\ldots\neq A^{5+}$ and $A^5=A^{6+}$, $R_p=A^{5+}$. Scanning R_p, we get the shortest distances between all points and the one from the original point to the terminal point directly (11.2).

2.4 Calculating SP of a Graph

A adjacent matrix A is shown as bellow:

$$A=\begin{pmatrix}0&19&\infty&19&24&28&\infty&\infty&\infty&\infty&\infty&\infty&\infty&\infty\\ 19&0&24&23&18&20&\infty&\infty&\infty&\infty&\infty&\infty&\infty&\infty\\ \infty&24&0&29&20&15&\infty&\infty&\infty&\infty&\infty&\infty&\infty&\infty\\ 19&23&29&0&\infty&\infty&16&17&25&\infty&\infty&\infty&\infty&\infty\\ 24&18&20&\infty&0&\infty&12&13&\infty&\infty&\infty&\infty&\infty&\infty\\ 28&20&15&\infty&\infty&0&27&15&\infty&\infty&19&\infty&\infty&\infty\\ \infty&\infty&\infty&16&12&27&0&\infty&11&15&23&\infty&\infty&\infty\\ \infty&\infty&\infty&17&13&15&\infty&0&26&18&13&\infty&\infty&\infty\\ \infty&\infty&\infty&25&\infty&\infty&11&26&0&\infty&\infty&26&9&21&28\\ \infty&\infty&\infty&\infty&\infty&\infty&15&18&\infty&0&\infty&26&14&10&17\\ \infty&\infty&\infty&\infty&\infty&19&23&13&\infty&\infty&0&29&21&16&10\\ \infty&\infty&\infty&\infty&\infty&\infty&\infty&\infty&26&26&29&0&20&\infty&\infty\\ \infty&\infty&\infty&\infty&\infty&\infty&\infty&\infty&9&14&21&20&0&13&\infty\\ \infty&\infty&\infty&\infty&\infty&\infty&\infty&\infty&21&10&16&\infty&13&0&18\\ \infty&\infty&\infty&\infty&\infty&\infty&\infty&\infty&28&17&10&\infty&\infty&18&0\end{pmatrix}$$

The elements on it are time between points communication of a transport network. In the calculating SP of a graph, Step 2 in Section 2.2 doesn't be needed, so the calculation is easier than that of a diagraph. The matrices responding to the example are gained according to the calculating process of SPM.

$$A^{2+} = \begin{pmatrix}
0 & 19 & 43 & 19 & 24 & 28 & 35 & 36 & 44 & \infty & 47 & \infty & \infty & \infty & \infty \\
19 & 0 & 24 & 23 & 18 & 20 & 30 & 31 & 48 & \infty & 39 & \infty & \infty & \infty & \infty \\
43 & 24 & 0 & 29 & 20 & 15 & 32 & 30 & 54 & \infty & 34 & \infty & \infty & \infty & \infty \\
19 & 23 & 29 & 0 & 28 & 32 & 16 & 17 & 25 & 31 & 30 & 51 & 34 & 46 & 53 \\
24 & 18 & 20 & 28 & 0 & 28 & 12 & 13 & 23 & 27 & 26 & \infty & \infty & \infty & \infty \\
28 & 20 & 15 & 32 & 28 & 0 & 27 & 15 & 38 & 33 & 19 & 48 & 40 & 35 & 29 \\
35 & 30 & 32 & 16 & 12 & 27 & 0 & 25 & 11 & 15 & 23 & 37 & 20 & 25 & 32 \\
36 & 31 & 30 & 17 & 13 & 15 & 25 & 0 & 26 & 18 & 13 & 42 & 32 & 28 & 23 \\
44 & 48 & 54 & 25 & 23 & 38 & 11 & 26 & 0 & 23 & 30 & 26 & 9 & 21 & 28 \\
\infty & \infty & \infty & 31 & 27 & 33 & 15 & 18 & 23 & 0 & 26 & 26 & 14 & 10 & 17 \\
47 & 39 & 34 & 30 & 26 & 19 & 23 & 13 & 30 & 26 & 0 & 29 & 21 & 16 & 10 \\
\infty & \infty & \infty & 51 & \infty & 48 & 37 & 42 & 26 & 26 & 29 & 0 & 20 & 33 & 39 \\
\infty & \infty & \infty & 34 & \infty & 40 & 20 & 32 & 9 & 14 & 21 & 20 & 0 & 13 & 31 \\
\infty & \infty & \infty & 46 & \infty & 35 & 25 & 28 & 21 & 10 & 16 & 33 & 13 & 0 & 18 \\
\infty & \infty & \infty & 53 & \infty & 29 & 32 & 23 & 28 & 17 & 10 & 39 & 31 & 18 & 0
\end{pmatrix}$$

$$A^{3+} = \begin{pmatrix}
0 & 19 & 43 & 19 & 24 & 28 & 35 & 36 & 44 & 50 & 47 & 70 & 53 & 63 & 57 \\
19 & 0 & 24 & 23 & 18 & 20 & 30 & 31 & 41 & 45 & 39 & 68 & 57 & 55 & 49 \\
43 & 24 & 0 & 29 & 20 & 15 & 32 & 30 & 43 & 47 & 34 & 63 & 55 & 50 & 44 \\
19 & 23 & 29 & 0 & 28 & 32 & 16 & 17 & 25 & 31 & 30 & 51 & 34 & 41 & 40 \\
24 & 18 & 20 & 28 & 0 & 28 & 12 & 13 & 23 & 27 & 26 & 49 & 32 & 37 & 36 \\
28 & 20 & 15 & 32 & 28 & 0 & 27 & 15 & 38 & 33 & 19 & 48 & 40 & 35 & 29 \\
35 & 30 & 32 & 16 & 12 & 27 & 0 & 25 & 11 & 15 & 23 & 37 & 20 & 25 & 32 \\
36 & 31 & 30 & 17 & 13 & 15 & 25 & 0 & 26 & 18 & 13 & 42 & 32 & 28 & 23 \\
44 & 41 & 43 & 25 & 23 & 38 & 11 & 26 & 0 & 23 & 30 & 26 & 9 & 21 & 28 \\
50 & 45 & 47 & 31 & 27 & 33 & 15 & 18 & 23 & 0 & 26 & 26 & 14 & 10 & 17 \\
47 & 39 & 34 & 30 & 26 & 19 & 23 & 13 & 30 & 26 & 0 & 29 & 21 & 16 & 10 \\
70 & 68 & 63 & 51 & 49 & 48 & 37 & 42 & 26 & 26 & 29 & 0 & 20 & 33 & 39 \\
53 & 57 & 55 & 34 & 32 & 40 & 20 & 32 & 9 & 14 & 21 & 20 & 0 & 13 & 31 \\
63 & 55 & 50 & 41 & 37 & 35 & 25 & 28 & 21 & 10 & 16 & 33 & 13 & 0 & 18 \\
57 & 49 & 44 & 40 & 36 & 29 & 32 & 23 & 28 & 17 & 10 & 39 & 31 & 18 & 0
\end{pmatrix}$$

$$A^{4+} = \begin{pmatrix}
0 & 19 & 43 & 19 & 24 & 28 & 35 & 36 & 44 & 50 & 47 & 70 & 53 & 60 & 57 \\
19 & 0 & 24 & 23 & 18 & 20 & 30 & 31 & 41 & 45 & 39 & 67 & 50 & 55 & 49 \\
43 & 24 & 0 & 29 & 20 & 15 & 32 & 30 & 43 & 47 & 34 & 63 & 52 & 50 & 44 \\
19 & 23 & 29 & 0 & 28 & 32 & 16 & 17 & 25 & 31 & 30 & 51 & 34 & 41 & 40 \\
24 & 18 & 20 & 28 & 0 & 28 & 12 & 13 & 23 & 27 & 26 & 49 & 32 & 37 & 36 \\
28 & 20 & 15 & 32 & 28 & 0 & 27 & 15 & 38 & 33 & 19 & 48 & 40 & 35 & 29 \\
35 & 30 & 32 & 16 & 12 & 27 & 0 & 25 & 11 & 15 & 23 & 37 & 20 & 25 & 32 \\
36 & 31 & 30 & 17 & 13 & 15 & 25 & 0 & 26 & 18 & 13 & 42 & 32 & 28 & 23 \\
44 & 41 & 43 & 25 & 23 & 38 & 11 & 26 & 0 & 23 & 30 & 26 & 9 & 21 & 28 \\
50 & 45 & 47 & 31 & 27 & 33 & 15 & 18 & 23 & 0 & 26 & 26 & 14 & 10 & 17 \\
47 & 39 & 34 & 30 & 26 & 19 & 23 & 13 & 30 & 26 & 0 & 29 & 21 & 16 & 10 \\
70 & 67 & 63 & 51 & 49 & 48 & 37 & 42 & 26 & 26 & 29 & 0 & 20 & 33 & 39 \\
53 & 50 & 52 & 34 & 32 & 40 & 20 & 32 & 9 & 14 & 21 & 20 & 0 & 13 & 31 \\
60 & 55 & 50 & 41 & 37 & 35 & 25 & 28 & 21 & 10 & 16 & 33 & 13 & 0 & 18 \\
57 & 49 & 44 & 40 & 36 & 29 & 32 & 23 & 28 & 17 & 10 & 39 & 31 & 18 & 0
\end{pmatrix}$$

$$A^{5+} = \begin{pmatrix} 0 & 19 & 43 & 19 & 24 & 28 & 35 & 36 & 44 & 50 & 47 & 70 & 53 & 60 & 57 \\ 19 & 0 & 24 & 23 & 18 & 20 & 30 & 31 & 41 & 45 & 39 & 67 & 50 & 55 & 49 \\ 43 & 24 & 0 & 29 & 20 & 15 & 32 & 30 & 43 & 47 & 34 & 63 & 52 & 50 & 44 \\ 19 & 23 & 29 & 0 & 28 & 32 & 16 & 17 & 25 & 31 & 30 & 51 & 34 & 41 & 40 \\ 24 & 18 & 20 & 28 & 0 & 28 & 12 & 13 & 23 & 27 & 26 & 49 & 32 & 37 & 36 \\ 28 & 20 & 15 & 32 & 28 & 0 & 27 & 15 & 38 & 33 & 19 & 48 & 40 & 35 & 29 \\ 35 & 30 & 32 & 16 & 12 & 27 & 0 & 25 & 11 & 15 & 23 & 37 & 20 & 25 & 32 \\ 36 & 31 & 30 & 17 & 13 & 15 & 25 & 0 & 26 & 18 & 13 & 42 & 32 & 28 & 23 \\ 44 & 41 & 43 & 25 & 23 & 38 & 11 & 26 & 0 & 23 & 30 & 26 & 9 & 21 & 28 \\ 50 & 45 & 47 & 31 & 27 & 33 & 15 & 18 & 23 & 0 & 26 & 26 & 14 & 10 & 17 \\ 47 & 39 & 34 & 30 & 26 & 19 & 23 & 13 & 30 & 26 & 0 & 29 & 21 & 16 & 10 \\ 70 & 67 & 63 & 51 & 49 & 48 & 37 & 42 & 26 & 26 & 29 & 0 & 20 & 33 & 39 \\ 53 & 50 & 52 & 34 & 32 & 40 & 20 & 32 & 9 & 14 & 21 & 20 & 0 & 13 & 31 \\ 60 & 55 & 50 & 41 & 37 & 35 & 25 & 28 & 21 & 10 & 16 & 33 & 13 & 0 & 18 \\ 57 & 49 & 44 & 40 & 36 & 29 & 32 & 23 & 28 & 17 & 10 & 39 & 31 & 18 & 0 \end{pmatrix}$$

3 Comparison between Approaches

The 'Dijkstra' and 'Bellman-Ford' approaches are similar to 'Floyd' approach. They are used to calculate the shortest paths between a single source and all the others, if the same process is done by changing sources n times, the shortest paths between all points can be figured out. So this paper focuses on comparison between 'Floyd' and SPM.

The idea of 'Floyd': there is a square matrix $\boldsymbol{A}^{(k)}$, the element $a^{(k)}[i][j]$ $(i \neq j)$ shows the distance from *i to j* , k expresses operating times[11-12] .

At the very start (k=-1), if there is an edge from i to j, $a^{(-1)}[i][j]$ is the weight of the edge, otherwise, it is substituted by '∞'. So the matrix $\boldsymbol{A}^{(-1)}$ of Fig.1 is gained as follows:

$$A^{(-1)} = Edge = \begin{pmatrix} 0 & 1 & \infty & \infty & \infty & \infty & \infty \\ \infty & 0 & 2.2 & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 3.5 & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 2 & 2 & \infty \\ \infty & \infty & 3 & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix}$$

Next, other points are added in the path of $a[i][j]$, if $a[i][k]+a[k][j] \prec a[i][j]$, $a[i][j]$ is substituted by $a[i][k]+a[k][j]$. That is, first, Point '0'is added in the path of $a[0][0]$ and a new $a[0][0]$ is gained, and then Point '0'is added in the path of $a[0][1]$, $a[0][2]$…, through n times, the shortest paths between '0'and the others through intermediate '0'; in a similar way, through n^2 times the second line to Line (n-1) of $\boldsymbol{A}^{(0)}$ are figured out as follows:

$$A^{(0)} = \begin{pmatrix} 0 & 1 & \infty & \infty & \infty & \infty & \infty \\ \infty & 0 & 2.2 & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 3.5 & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 2 & 2 & \infty \\ \infty & \infty & 3 & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix}$$

Next, Point '1', '2'… and '*n-1*' are introduced in turn. Through n^3 times, $\boldsymbol{A}^{(1)}$, $\boldsymbol{A}^{(2)}$…, $\boldsymbol{A}^{(n-1)}$ (which show the paths between all points through intermediate points '0'～'*n-1*' in turn) are gained:

$$A^{(1)} = \begin{pmatrix} 0 & 1 & 3.2 & \infty & \infty & \infty & \infty \\ \infty & 0 & 2.2 & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 3.5 & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 2 & 2 & \infty \\ \infty & \infty & 3 & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix} \quad A^{(2)} = \begin{pmatrix} 0 & 1 & 3.2 & 6.7 & \infty & \infty & \infty \\ \infty & 0 & 2.2 & 5.7 & \infty & \infty & \infty \\ \infty & \infty & 0 & 3.5 & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 2 & 2 & \infty \\ \infty & \infty & 3 & 6.5 & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix}$$

$$A^{(3)} = \begin{pmatrix} 0 & 1 & 3.2 & 6.7 & 8.7 & 8.7 & \infty \\ \infty & 0 & 2.2 & 5.7 & 7.7 & 7.7 & \infty \\ \infty & \infty & 0 & 3.5 & 5.5 & 5.5 & \infty \\ \infty & \infty & \infty & 0 & 2 & 2 & \infty \\ \infty & \infty & 3 & 6.5 & 0 & 8.5 & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix} \quad A^{(4)} = \begin{pmatrix} 0 & 1 & 3.2 & 6.7 & 8.7 & 8.7 & \infty \\ \infty & 0 & 2.2 & 5.7 & 7.7 & 7.7 & \infty \\ \infty & \infty & 0 & 3.5 & 5.5 & 5.5 & \infty \\ \infty & \infty & 5 & 0 & 2 & 2 & \infty \\ \infty & \infty & 3 & 6.5 & 0 & 8.5 & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix}$$

$$A^{(5)} = \begin{pmatrix} 0 & 1 & 3.2 & 6.7 & 8.7 & 8.7 & 11.2 \\ \infty & 0 & 2.2 & 5.7 & 7.7 & 7.7 & 10.2 \\ \infty & \infty & 0 & 3.5 & 5.5 & 5.5 & 8 \\ \infty & \infty & 5 & 0 & 2 & 2 & 4.5 \\ \infty & \infty & 3 & 6.5 & 0 & 8.5 & 11 \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix} \quad A^{(6)} = \begin{pmatrix} 0 & 1 & 3.2 & 6.7 & 8.7 & 8.7 & 11.2 \\ \infty & 0 & 2.2 & 5.7 & 7.7 & 7.7 & 10.2 \\ \infty & \infty & 0 & 3.5 & 5.5 & 5.5 & 8 \\ \infty & \infty & 5 & 0 & 2 & 2 & 4.5 \\ \infty & \infty & 3 & 6.5 & 0 & 8.5 & 11 \\ \infty & \infty & \infty & \infty & \infty & 0 & 2.5 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix}$$

According to the above, the 'Floyd' approach is complicated, n^3 times calculation must be done for $\boldsymbol{A}^{(n-1)}$. The large the n is, the lower the efficiency is and there are duplicate calculations $(\boldsymbol{A}^{(-1)}=\boldsymbol{A}^{(0)}, \boldsymbol{A}^{(5)}=\boldsymbol{A}^{(6)})$.

Compared to the 'Floyd' approach, SPM have following advantages:

1. It converts calculation of the elements to standard matrix operation. Through the first times operation, all the shortest paths between all points through one intermediate point (or two paths) are get. That is one times Multiply-add operation can complete function of inner times n^2 calculation of the 'Floyd' approach;

2. The example 2.3 shows that the operation of SPM terminate when $\boldsymbol{A}^{r+}=\boldsymbol{A}^{(r-1)+}$ and don't need n times;

3. If there are no change between the values of corresponding line of $\boldsymbol{A}^{(r-1)+}$and $\boldsymbol{A}^{r+}$, the line is copied as the corresponding line of $\boldsymbol{A}^{(r+1)+}$, so duplicate calculations are

reduced. (The values of Line *n-2* on $\boldsymbol{A}^{2+}$ and $\boldsymbol{A}$ are the same in Section 2.3, the corresponding values on $\boldsymbol{A}^{3+}$ don't be needed to calculate.).

According to the above, the more the points are and the less the difference of distance between points, the less the times of SPM are needed (There are 15 points, only 5 times calculations are needed and all the shortest paths between all the points are gained).

4 Conclusions

A new approach for the shortest paths is proposed. The approach introduce the idea of the reachable matrix into calculation of the shortest paths, provided a new solution. The respective descriptions and rules are also given in this paper. In the end, through compared to the 'Floyd' approach, the advantages of SPM are highlight.

References

1. Sedgewick, R., Flajoler, P.: An Introduction to the Analysis of Algorithms. Addison-Wesley Professional, USA (2006)
2. Sipser, M.: Introduction to the Theory of Computation, 2nd edn. Thomson, USA (2006)
3. Yu, D., Zhang, Q., Ma, S., et al.: Optimized Dijkstra Algorithm. J. Computer Engineering 30(22), 145–146 (2004)
4. Guha, S., Kettinger, W.J., Teng, J.T.C.: Business Process Reengineering Building a Comprehensive Methodology. J. Information System Management, 13–22 (1993)
5. Zhang, Z.-h., Wang, D.-w.: BPR-Based Business Operation Procedure in Electronic Brokering. J. Journal of Northeastern University (Natural Science) 26(11), 1029–1032 (2005)
6. Wang, L.: Systems engineering theory methods and applying, pp. 35–40. Higher Education Press, Beijing (2004)
7. Steward, D.V.: The design structure system: A method for managing the design of complex systems. J. IEEE Trans. Engineering Management 78(3), 71–74 (1981)
8. Wang, D., Tang, Z., Ip, W.H., et al.: A Human-Computer Interactive Approach based on Activity-Section Analysis for BPR. J. Production Planning and Control 11(8), 789–796 (2000)
9. Xu, L.-n., Zhang, H.-m., Xu, W.-s., et al.: Application of DSM-based Process Reengineering in Multidisciplinary Cooperative Design. In: The 9th International Conference on Computer Supported Cooperative Work in Design Proceedings, pp. 961–965 (2005)
10. Cho, S.-H., Eppinger, S.D.: A Simulation-Based Process Model for Managing Complex Design Projects. IEEE Trans. Engineering Management 52(3), 316–328 (2005)
11. Tian, J., Gui, L., Li, Z.-g.: A approach for transport problem combining the shortest path method. Highways & Automotive Applications 3(125), 57–58 (2008)
12. Weimin, Y., Weimin, W.: Data Structure. Tsinghua University Press, Beijing (2002)

A Particle Swarm Optimization Heuristic for the Index Tacking Problem

Hanhong Zhu[1], Yun Chen[2], and Kesheng Wang[3]

[1,2] School of Public Economics & Administration,
Shanghai University of Finance and Economics (SUFE), Shanghai 200433, China
[3] Department of Production and Quality Engineering,
Norway University of Science and Technology (NTNU), Trondheim 7491, Norway
zhh@mail.shufe.edu.cn

Abstract. Considering the market is efficient, an obvious portfolio management strategy is passive where the challenge is to track a certain benchmark like a stock index such that equal returns and risks are achieved. An index tracking problem is to minimize the tracking error between a portfolio and a certain benchmark. In this paper, we present a heuristic approach based on particle swarm optimization (PSO) techniques to optimize the solution of the index tracking problem. Our objective is to replicate the performance of a given portfolio under the condition that the number of stocks allowed in the portfolio is smaller than the number of stocks in the benchmark index. In order to evaluate the performance of PSO, the results in this study has been used to compare with those obtained by the genetic algorithms (GAs). The computational results show that particle swarm optimization approach is efficient and effective for solving index tracking optimization problems and the performance of PSO is better than GAs.

Keywords: Particle swarm optimization, Index tracking, Track error, Passive investment management.

1 Introduction

With Difference from traditional evolutionary computation techniques, a new heuristic approach, called particle swarm optimization (PSO) has been introduced recently by Kennedy and Eberhart[1]. PSO is motivated from the simulation of social behavior. Many researches on applying heuristic algorithms in portfolio optimization have been reported in literatures, but none of them applies the PSO heuristic for solving index tracking problems. This paper presents a new approach to solve index tracking problem using PSO.

Active and passive management are two main strategies in investment management. Because active strategies rely heavily on skillful investors who can manage and control the market, it is difficult to be implemented in practice. In recent years, passive investment strategies have become very popular, especially among mutual fund managers and pension funds. Passive strategies are adopted by investors who believe

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 238–245, 2010.

that financial markets are efficient and it is impossible to consistently beat the aggregate market return.

Index tracking is one of passive strategies, which attempts to reproduce the performance of a theoretical index representing the market as closely as possible. A realistic formulation of the problem should include the number of assets in the portfolio, the restrictions on the positions on each asset, the size of transactions costs, as well as liquidity and exposure constraints. The effective method of handling them in the context of a realistic problem size is to use heuristic algorithms that provide good approximations of the optimal solution[2].

Some researchers investigated the index tracking optimization, in which heuristic optimization technique, as a new direction of research, has been applied for the solution of the index tracking problem. For example, Threshold Accepting Heuristic (TA)[3], Simulated Annealing (SA), Tabu Search (TS)[4], Genetic Algorithm (GA)[5] and Ant Colony Optimization (ACO)[6]. The advantage of these techniques is that virtually no restriction has to be imposed on the shape of the objective function or the constraints.

In this paper we propose another heuristic optimization algorithm, called Particle Swarm Optimization (PSO). PSO has been successfully applied for solving portfolio optimization problems[7]. Following the results obtained in our experiments in the paper, we demonstrate PSO is an efficient technique for the index tracking. The results of the PSO study are also compared with those of genetic algorithm (GA) and the PSO shows a better performance than the GA.

The remainder of the paper is organized as follows. Section 2 presents the optimization problem of index tracking. Section 3 describes the general concept of PSO and how to using PSO to solve the index tracking problem. Section 4 shows and discuss the results in the experiment. Conclusions are drawn in Section 5.

2 The Optimization Problem

The optimization problem of index tracking is to find an optimal value of reproducing the performance of a given index. The desired solution is a portfolio which is composed of a relatively small subset of the stocks in the market and behaves similar to the index. This section describes the mathematical model of the problem. The most equations of the model are based on the work of Beasley et al.[5].

2.1 Objective Function

We suppose that there are *N* assets in the market, from which a tracking portfolio should be found. Let p_{it} be the price at time *t* of the asset *i*, *i* = 0,…,*N*. Let I_t be the value of the index observed at time *t*. We have observed the market prices p_{it} and I_t for periods t_0,…, t_1 in the past. We attempt to find the composition of a portfolio, which would have tracked in an optimal way the index over the period $[t_0, t_1]$. In other words, we attempt to find quantities w_{it}, i = 1,…,*N*, and an objective value

F_{t_0,t_1}. F will be a function of the tracking error of the tracking portfolio against the index value I_t. The measure used for the tracking error in returns over the period of time between t_0 and t_1 is E_{t_0,t_1} defined as

$$E_{t_0,t_1} = \frac{(\sum_{t=t_0}^{t_1} \left| r_t^P - r_t^I \right|^{\alpha})^{1/\alpha}}{t_1 - t_0} \tag{1}$$

Where α>0. r_t^P and r_t^I are the return on the tracking portfolio and index over the period [t_0, t_1] respectively. Especially $r_t^P = \sum_{i=1}^{N} w_{it} r_{it}$.

Let R_{t_0,t_1} be the average of the deviations of the tracking portfolio returns from the index returns over the period of time [t_0, t_1],

$$R_{t_0,t_1} = \frac{\sum_{t=t_0}^{t_1} (r_t^P - r_t^I)}{t_1 - t_0} \tag{2}$$

Positive deviations from the index may be desirable. One way to account for this is to define the function to be minimized F_{t_0,t_1} as a weighted difference of our measure of the tracking error E_{t_0,t_1} and of the excess return R_{t_0,t_1}:

$$F_{t_0,t_1} = \lambda E_{t_0,t_1} - (1-\lambda) R_{t_0,t_1} \tag{3}$$

Where $\lambda \in [0,1]$. The sensitivity of the investor to the risk increases as λ approaches unity, while it decreases as λ approaches zero.

2.2 Problem Formulation

Our problem consists in finding a new portfolio $P_{t_0} = \{ w_{it_0} \}$ which, if we hold it unchanged during the period [t_0, t_1], minimizes the objective function F_{t_0,t_1} under the constraints:

$$\underset{P_{t_0}}{Min}\, F_{t_0,t_1} = \lambda E_{t_0,t_1} - (1-\lambda) R_{t_0,t_1} \tag{6}$$

Subject to

$$\sum_{i=1}^{N} w_{it_0} = 1,\ w_{it_0} \geq 0 \tag{7}$$

$$\sum_{i=1}^{N} z_{it_0} = K \tag{8}$$

$$\varepsilon_{it_0} z_{it_0} \leq w_{it_0} \leq \delta_{it_0} z_{it_0},\ \text{i=1,…,N} \tag{9}$$

$$z_{it_0} \in \{0,1\}\ \text{ i=1,…,N.} \tag{10}$$

Where Eq.(7) means that no short sales are allowed. In deed, it is a common practice to sell assets that are not yet owned by the investor at the time, in expectation of falling prices. This can be formulated by replacing $\sum_{i=1}^{N} w_{it_0} = 1$, $w_{it_0} \geq 0$ with $\sum_{i=1}^{N} w_{it_0} = 1$. This relaxation was introduced by[8]. Eq.(8) is Cardinality constraints, and Eq.(9) is Size constraints, This constraints are introduced in an index tracking framework by[9]. Eq.(10) is asset selection constraints, if z_{it_0} equal 1, the stock i will be selected in the portfolio, otherwise the stock i will be excluded.

By solving Eq. (6), we can find the number of each asset *i* that we should have held during all the period $[t_0, t_1]$ in the past in order to optimally reproduce the performance of the index during the same period. Assuming that the same portfolio will also optimally track the index during period $[t_1, t_2]$ in the future, at time t_1 we use the weights $w_{i,t_1} = w_{i,t_0}$ to construct a portfolio P_{t_1} in which the weight invested in asset *i* is the same as the corresponding weight in the portfolio P_{t_0}.

3 PSO for Optimization of Index Tracking

There is no any accurate numerical solution for Eqs. (6)-(10) for the reason that it is a mixed quadratic and integer programming problem[10]. Hence in order to solve the index tracking problem, we introduce PSO heuristic approach. It is one of the evolutionary optimization methods and is based on the metaphor of social interaction and communication behavior of biological swarms, such as bird flocking.

PSO could be explained well using a scenario: a group of birds are flying in an area to look for food, the easiest way to find the food is to follow the one who is closest to the food. In other words, Any particle moves to find the best solution by following its own best experience and the swarm's best experience.

We can consider that there should be *M* particles in the swarm, each representing a portfolio including *N* assets. Each particle includes proportion variables denoted by w_{pi} (p=1,…,M, where p is the portfolio in the solution space. i=1,…,N, where w_{pi} is the proportion of every asset in the portfolio). The best particle is the optimal portfolio we wanted.

3.1 Fitness Function

Each particle in the PSO swarm will be associated with a fitness value. Thus, in the swarm, particles' best experience is the position where the particle has met with the best fitness value, and the neighbor's best experience is the previous position where the neighbor has met the best fitness value. In this paper, we select the fitness function as follow:

$$f_P = \lambda E_{t_0,t_1} - (1-\lambda) R_{t_0,t_1} \tag{11}$$

Where f_P is the fitness value of particle p. Every particle of the swarm represents a candidate solution, and each particle must be feasible and satisfy Eqs. (7)-(10). In this paper, about the cardinality constraints, we set the K equal 15. About size constraints, we set the parameter as: ε_i =0, and δ_i =1.

3.2 PSO Updating

In the algorithm of PSO, each solution is called a "particle", and every particle has its position, velocity, and fitness value. At each iteration, every particle moves towards its personal best position and towards the best particle of the swarm found so far. The velocity changes according to Eq. (12):

$$\vec{v}_i(t+1) = w\vec{v}_i(t) + c_1 r_1[\vec{p}_i(t) - \vec{x}_i(t)] + c_2 r_2[\vec{p}_g(t) - \vec{x}_i(t)] \tag{12}$$

where t is the iteration sequence of the particle i. c_1 and c_2 are positive constant parameters called acceleration coefficients, which are responsible for controlling the maximum step size. r_1 and r_2 are random numbers between (0, 1). w is a constant. $\vec{v}_i(t+1)$ is particle i's velocity at iteration t + 1. $\vec{v}_i(t)$ is particle i's velocity at iteration t. $\vec{x}_i(t)$ is particle i's position at iteration t. $\vec{p}_i(t)$ is the historical best position of particle i. $\vec{p}_g(t)$ is the historical best position of the swarm.

The new position of particle i, $\vec{x}_i(t+1)$, is calculated by Eq. (13)

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1) \tag{13}$$

The details about PSO algorithm can be referred to the paper[11].

4 Computational Experiments

In the paper, we selected portfolios consist of 15 stocks (equal K of eq. (8)) from S&P 500 index in USA, FTSE 100 index in UK and SSE 50 index in China. the percentages $p1$, $p2$, ..., $p15$ of our investment cash need to be computed, so that we are able to determine how much should be invested in each of the 15 stocks. In this section, we present the experimental results obtained when searching the best portfolio, which provides the solution of the problem using Eqs.(6)-(10). In order to evaluate the performance of PSO model, we have also compared the PSO approach with another heuristic approach genetic algorithm (GA)[5]. The experiments data were obtained from website: http://finance.yahoo.com. These data correspond to weekly prices between January 2007 and December 2007 from the three indices. In the experiments, PSO Solver has been developed using Matlab. Genetic Algorithm has been developed using GeneHunter[12].

All the results were computed using the α =2 and $\Delta\lambda$ =0.2 for the implementation of the algorithms. The PSO and GA algorithms used the same test data and were run on the same computer. The results are shown in the table 1. The table 1 shows the tracking error, excess return and fitness value obtained by GA and PSO heuristics

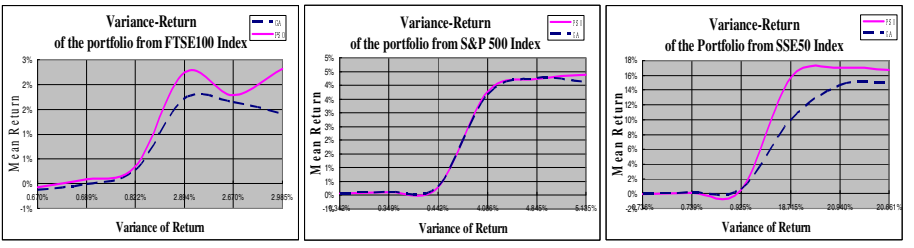
respectively. All the fitness values obtained by PSO heuristic are always less than them obtained by GA heuristic. So we can conclude that PSO heuristic is superior to GA heuristic in solving the index tracking problem.

Taking the sets of optimal portfolios obtained with each heuristic, we trace out their Variance-Return curve shown in Fig.1. Fig.2(a)-(c) show the comparison of the Variance-Return curve by GA and PSO heuristics from the three Index. The solid line represents the curve got by PSO heuristics and the dotted line represents the curve got by GA. The solid lines are always above the dotted lines clearly. The fact shows that the PSO heuristic is superior to genetic algorithm.

The tracking FTSE100 Index curves obtained by PSO and GA heuristics are shown in Figs.2 (a)-(f). We can draw two conclusions by observation. First, the tracking curves obtained by PSO are closer to the index than the curves obtained by GA. Second, the value of λ is very important to solve the optimization of the index tracking problem. In this case, we can get the closest tracking curve by selecting $\lambda = 1$.

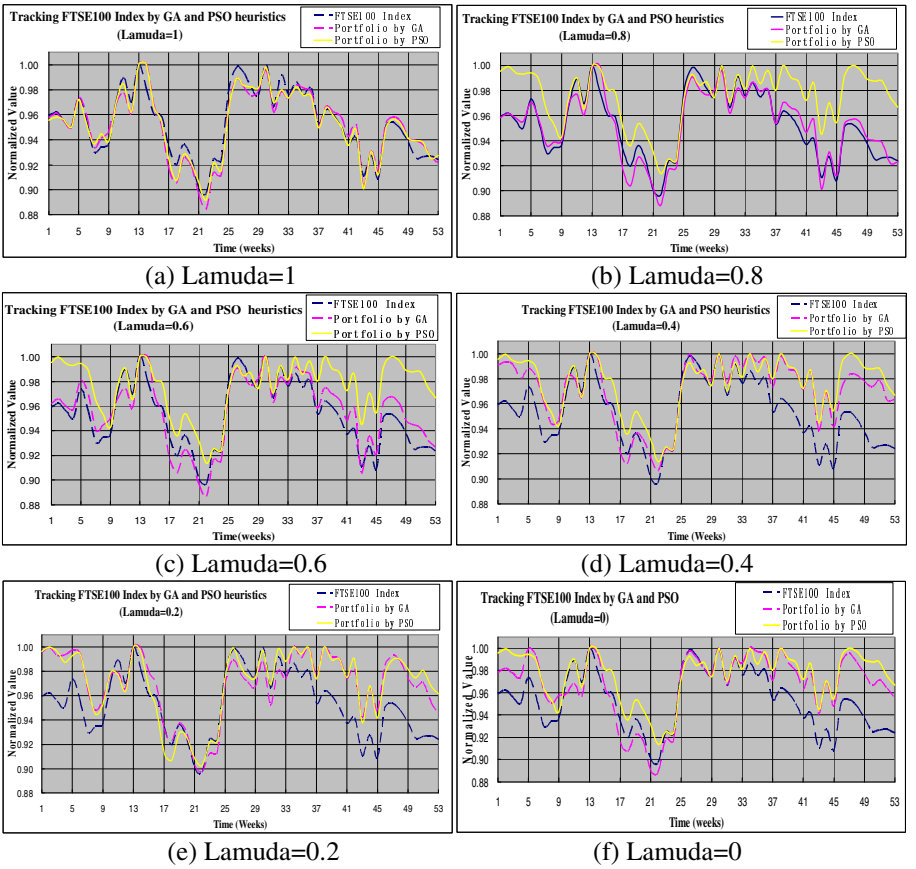
Table 1. The experimental results of PSO and GA heuristics

Index	λ	Tracking error		Excess return		Fitness Value	
		GA	PSO	GA	PSO	GA	PSO
FTSE 100	1	0.814%	0.670%	-0.129%	-0.069%	0.814%	0.670%
	0.8	0.786%	0.689%	-0.018%	0.095%	0.632%	0.532%
	0.6	1.019%	0.822%	0.271%	0.351%	0.503%	0.353%
	0.4	2.413%	2.894%	1.717%	2.247%	-0.065%	-0.191%
	0.2	2.564%	2.670%	1.638%	1.785%	-0.798%	-0.894%
	0	2.384%	2.985%	1.408%	2.309%	-1.408%	-2.309%
S&P 500	1	0.360%	0.342%	0.058%	0.054%	0.360%	0.342%
	0.8	0.367%	0.349%	0.112%	0.120%	0.271%	0.255%
	0.6	0.459%	0.442%	0.310%	0.299%	0.152%	0.145%
	0.4	3.964%	4.086%	3.675%	3.761%	-0.619%	-0.622%
	0.2	4.958%	4.845%	4.249%	4.244%	-2.408%	-2.426%
	0	4.700%	5.135%	4.099%	4.398%	-4.099%	-4.398%
SSE 50	1	0.795%	0.736%	-0.045%	-0.054%	0.795%	0.736%
	0.8	0.827%	0.739%	0.163%	0.133%	0.629%	0.565%
	0.6	1.026%	0.925%	0.592%	0.553%	0.379%	0.333%
	0.4	11.866%	18.715%	10.016%	15.722%	-1.263%	-1.947%
	0.2	17.860%	20.940%	14.658%	17.065%	-8.154%	-9.464%
	0	18.323%	20.661%	14.959%	16.741%	-14.96%	-16.74%



a) FTSE100 Index (b) S&P500 Index (c) SSE50 Index

Fig. 1. Variance-Return curves of the portfolio by PSO and GA heuristics



(a) Lamuda=1 (b) Lamuda=0.8

(c) Lamuda=0.6 (d) Lamuda=0.4

(e) Lamuda=0.2 (f) Lamuda=0

Fig. 2. Tracking FTSE100 index curve by GA and PSO heuristics

5 Conclusion

We present the index tracking problem and develop a PSO heuristic approach for its solution in this paper. In the experiments, the results obtained by PSO heuristic and

another heuristic GA algorithm have been compared with. The results show that PSO heuristic is superior to solving the index tracking problem than GA heuristic in this case. The structure of PSO is simpler and computational time is shorter. So we can come to conclude that PSO is an effective and efficient approach to the passive portfolio management. In order to solve the index tracking in real world, the further research will be focus on how can apply more complex portfolio optimization model to the index tracking problem.

Acknowledgments. The authors thank Dr. Yi Wang (Nottingham Business School, NTU, UK) and two reviewers for their helpful comments.

References

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: International Conference on Neural Networks, pp. 1942–1948 (1995)
2. Gilli, M., Winker, P.: A Review of Heuristic Optimization Methods in Econometrics. Swiss Finance Institute (2008)
3. Gilli, M., Kellezi, E.: The Threshold Accepting Heuristic for Index Tracking. SSRN eLibrary (2001)
4. Tollo, G.d., Maringer, D.: Metaheuristics for the Index Tracking Problem. In: Metaheuristics in the Service Industry, pp. 127–154 (2009)
5. Beasley, J.E., Meade, N., Chang, T.J.: An Evolutionary Heuristic for the Index Tracking Problem. European Journal of Operational Research 148, 621–643 (2003)
6. Doerner, K.F., Gutjahr, W.J., Hartl, R.F., Strauss, C., Stummer, C.: Pareto Ant Colony Optimization with ILP Preprocessing in Multiobjective Project Portfolio Selection. European Journal of Operational Research 171, 830–841 (2006)
7. Cura, T.: Particle Swarm Optimization Approach to Portfolio Optimization. Nonlinear Analysis: Real World Applications 10, 2396–2406 (2009)
8. Black, F.: Capital Market Equilibrium with Restricted Borrowing. The Journal of Business 45, 444–455 (1972)
9. Derigs, U., Nickel, N.-H.: On a Local-Search Heuristic for a Class of Tracking Error Minimization Problems in Portfolio Management. Annals of Operations Research 131, 45–77 (2004)
10. Fernandez, A., Gomez, S.: Portfolio Selection Using Neural Networks. Computers & Operations Research 34, 1177 (2007)
11. Bratton, D., Kennedy, J.: Defining a Standard for Particle Swarm Optimization. In: Swarm Intelligence Symposium, pp. 120–127. IEEE, Los Alamitos (2007)
12. Wang, K.: Applied computational intelligence in intelligent manufacturing systems. In: Advanced Knowledge International, Adelaide, S. Aust., vol. 2 (2005)

Structural Design of Optimized Polynomial Radial Basis Function Neural Networks

Young-Hoon Kim, Hyun-Ki Kim, and Sung-Kwun Oh

Department of Electrical Engineering, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
ohsk@suwon.ac.kr

Abstract. In this paper, we introduce optimization methods of Polynomial Radial Basis Function Neural Network (pRBFNN). The connection weight of proposed pRBFNN is represented as four kinds of polynomials, unlike in most conventional RBFNN constructed with constant as connection weight. Input space in partitioned with the aid of kernel functions and each kernel function is used Gaussian type. Least Square Estimation (LSE) is used to estimate the coefficients of polynomial. Also, in order to design the optimized pRBFNN model, center value of each kernel function is determined based on C-Means clustering algorithm, the width of the RBF, the polynomial type in the each node, input variables are identified through Particle Swarm Optimization (PSO) algorithm. The performances of the NOx emission process of gas turbine power plant data and Automobile Miles per Gallon (MPG) data was applied to evaluate proposed model. We analyzed approximation and generalization of model.

Keywords: Polynomial Radial Basis Function Neural Networks (pRBFNN), C-Means, Particle Swarm Optimization Algorithm, Machine Learning data.

1 Introduction

Dimensionality issues have emerged as an important topic in neurocomputing given our ultimate challenge to deal with real-world problems of high dimensionality [1], [2]. Given the simple topological structure and universal approximation abilities [3], radial basis function neural networks (RBFNNs) have been widely studied and applied to many categories of problems such as those arising in pattern recognition, signal processing, time-series prediction, and nonlinear system modeling and control [4],[5],[6].

Clustering algorithm is the process of dividing data elements into classes or clusters and it widely used to extract information granule from data. C-Means clustering (HCM) and Fuzzy C-Means clustering (FCM) are used to get information granule (center of clusters) for generating spaces of fuzzy sets, Mountain clustering and subtractive clustering [7], [8] are used to automatically determine the number of rules. There are many studies related to the identification of fuzzy model using evolutionary optimization such as Genetic Algorithm (GA) and PSO [9].

The pRBFNN designed with the aid of HCM and the least square method involves structural as well as parametric optimization. As far as the structure optimization is concerned, there are three components to consider, i.e., a collection of specific subsets

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 246–253, 2010.

of the input variable, the width of the RBFs, and the order of the polynomials type of each node. Also the pRBFNN can have different types of polynomial one another. These three components impact the performance of the pRBFNN and have to be optimized. The PSO algorithm was exploited to maximize the accuracy of the pRBFNN.

In this study, the polynomial type of the consequent part of fuzzy rules is determined by the proposed algorithm. We optimized such parameters as a collection of specific subsets of the input variable, polynomial type in the consequent part of fuzzy rules and the width of RBF by using the PSO.

2 General RBF Neural Network

A RBFNN is an artificial neural network that uses RBFs as activation functions. It is a linear combination of RBFs. They are used in function approximation, time series prediction, and control [7].

As the RBF, thin plate type, cubic type, and linear type are used. Here, Gaussian function is considered, therefore, $R_j(x)$ denotes the premise part of fuzzy rules, and is given as the following form:

$$R_j(x) = e^{-\frac{\|X - v_j\|^2}{2(\sigma)^2}} \tag{1}$$

Eq. (1) can be represented as Eq. (2) and as kernel function, all RBFs in the hidden layer have same width as shown in the following form.

$$\begin{aligned} R_j(x) &= e^{-\left(\frac{(X_1 - v_{j1})^2}{2(\sigma)^2} + \frac{(X_2 - v_{j2})^2}{2(\sigma)^2} + \ldots + \frac{(X_k - v_{jk})^2}{2(\sigma)^2}\right)} \\ &= e^{-\frac{1}{2}\left(\frac{(X_1 - v_{j1})^2}{(\sigma)^2} + \frac{(X_2 - v_{j2})^2}{(\sigma)^2} + \ldots + \frac{(X_k - v_{jk})^2}{(\sigma)^2}\right)} \\ & e^{-\frac{1}{2}\left(\sum_{k=1}^{n} \frac{(X_k - v_{jk})^2}{(\sigma)^2}\right)} \end{aligned} \tag{2}$$

Where, X_k is input variables of $k(j=1,\ldots,l)^{\text{th}}$, v_{jk} denotes the center value of RBF of the $k^{\text{th}}(k=1,\ldots,n)$ input, σ is given as the width of RBF to determine activation area.

In the hidden layer, the center value and the width of the kernel function give significant impact on the output of the model.

3 Proposed Polynomial RBF Neural Network

3.1 Structure of Proposed RBFNN

In the proposed pRBFNN, the structure of kernel function is decided by the center values of RBFs and the widths of RBFs.

In this paper, the connection weight of proposed pRBFNN is represented as four kinds of polynomials unlike in most conventional RBFNN constructed with constant as connection weight.

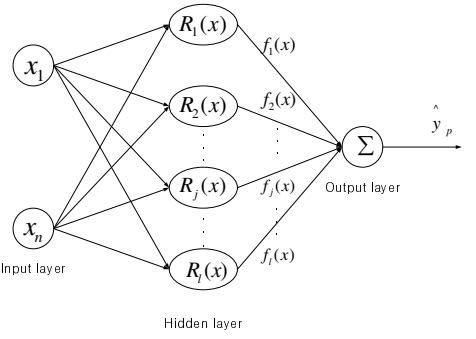


Fig. 1. Structure of the proposed RBFNN

[Case 1] First, in individual hidden nodes of RBF, the distribution constant (the width of RBF : σ) corresponding to each node is different one another.

$$R_j(x) = e^{-\left(\frac{(X_1-v_{j1})^2}{2(\sigma_j)^2}+\frac{(X_2-v_{j2})^2}{2(\sigma_j)^2}+\ldots+\frac{(X_k-v_{jk})^2}{2(\sigma_j)^2}\right)} \\ e^{-\frac{1}{2}\left(\sum_{k=1}^{n}\frac{(X_k-v_{jk})^2}{(\sigma_j)^2}\right)} \tag{3}$$

[Case 2] Second, the distribution constant (σ) corresponding to each input variable within individual hidden nodes (rules) is not the same.

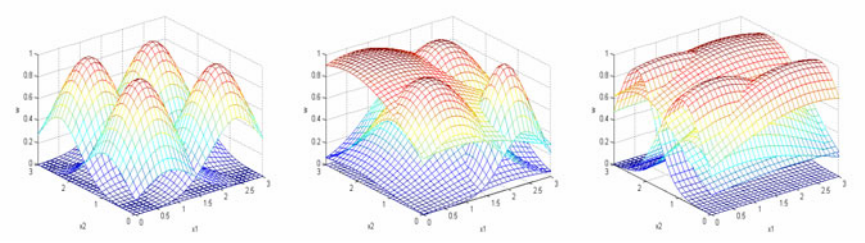
Fig.2-(a) shows standard RBF that the width of RBFs is all the same, Fig.2-(b) shows the width of RBF is different for each node as Case 1, and Fig.2-(c) shows the width of RBF is different for individual inputs per node as Case2.

$$R_j(x) = e^{-\left(\frac{(X_1-v_{j1})^2}{2(\sigma_{j1})^2}+\frac{(X_2-v_{j2})^2}{2(\sigma_{j2})^2}+\ldots+\frac{(X_k-v_{jk})^2}{2(\sigma_{jk})^2}\right)} \\ e^{-\frac{1}{2}\left(\sum_{k=1}^{n}\frac{(X_k-v_{jk})^2}{(\sigma_{jk})^2}\right)} \tag{4}$$

In the Fig. 1, $f_j(x)$ is a distinguished polynomial between the related polynomials $\hat{y}_p$ has the following form:

$$\hat{y}_p = \frac{\sum_{j=1}^{l}\left(R_j \cdot f_j\right)}{\sum_{j=1}^{l} R_j} = \sum_{j=1}^{l} w_j f_j \tag{5}$$

Where, $w_j = R_j / \sum_{i=1}^{j} R_i$ is the j^{th} normalized output of RBF, f_j is the j^{th} polynomial function of connection weight.



(a) Basic form of RBF (b) Case1 RBF (c) Case2 RBF

Fig. 2. Comparison of Radial Basis Function

3.2 Optimization of Proposed RBFNN

The pRBFNN designed with the aid of HCM and the LSE involves structural as well as parametric optimization. As far as the structure optimization is concerned, there are three components to consider, i.e. a collection of specific subsets of the input variable, the width of RBF and the order of polynomial. These three components to be optimized give impact on the performance of the pRBFNN. The PSO algorithm for optimization is exploited to improve the approximation as well as generalization capabilities of the pRBFNN model.

Fig. 3 shows the structure of particle of PSO used in the optimization of the pRBFNN.

[CASE 1] Fig. 3-(a) shows the first case of particle structure to decide the structure of pRBFNN. Individual hidden nodes with different distribution constant (the width of RBF: σ) are constructed one another for optimization of particle structure of PSO.

[CASE 2] Fig. 3-(b) shows the second case of particle structure to decide the structure of pRBFNN. It is the extended form of the first case. The particle structure of PSO is different for each input variable within individual hidden nodes (rules)

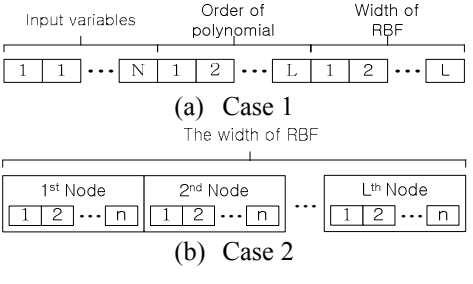


Fig. 3. Composition of particle structure

3.3 Object Function of PSO

The objective function (performance index) is a basic instrument guiding the evolutionary search in the solution space. The objective function is handled by considering

two cases of the training and testing data (two split way) or training, validation and testing data (three way split).

$$\text{objectfunction} = \text{PI} \times \theta + \text{EPI} \times (1 - \theta) \tag{6}$$

Here, PI, EPI (or VPI) denote the performance index for the training, testing (validation) data, respectively. Moreover θ is a weighting factor that allows us to strike a meaningful balance between the performance of the model for the training and testing (validation) data. Depending upon the values of the weighting factor, several specific cases of the objective function are worth distinguishing.

4 Experimental Results

The proposed architecture of the network, its development and resulting performance are illustrated with the aid of a series of numeric experiments. The first one is a NOx emission process data of gas turbine power plant. The second series of experiments is concerned with a selected data set from Machine Learning repository (http://archive.ica.usi.edu/ml/). We use the standard performance index of the Mean Square Error (MSE) and Root Mean Square Error (RMSE) as expressed by (7) and (8).

$$\text{MSE} = \frac{1}{m} \sum_{i=1}^{m} (y_p - \hat{y}_p)^2 \tag{7}$$

$$\text{RMSE} = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (y_p - \hat{y}_p)^2} \tag{8}$$

4.1 NOx Emission Process

NOx emission process is also modeled by using the data of gas turbine power plants. Till now, NOx emission processor is almost based on “standard” mathematical model in order to obtain regulation data from control process.

The input variables include ambient temperature (AT) at site, compressor speed (CS), low pressure turbine speed (LPTS), compressor discharge pressure (CDP), and turbine exhaust temperature (TET). The output variable is NOx. We consider 260 pairs of the original input-output data. The performance index is defined by (12).

The identification error (performance index) of the proposed model is much lower (superb) in comparison with some other models studied previously as shown in Table 1. The performance of the propose model depends on the polynomial type, the number of clusters and inputs. Moreover, according of the change of the number of parameters corresponding to the change of polynomial type, balance between performance and network complexity leads to the effective design of network. Moreover the increase of the number of clusters does not necessarily mean the decrease of identification error of network as shown in Table 1.

Table 1. Comparison of identification error with previous models

Model			Polynomial type of individual rules		No. of clusters /inputs	No. of parameters	PI	VPI	EPI
Regression model			2 (linear)		-	6	17.68	-	19.23
FNN[10](Gas+complex)			1	θ=0.4	30/5	-	6.269	-	8.778
			2	θ=0.4	30/5	-	3.725	-	5.291
Multi-FNN[11]			2	θ=0.75	30/5	-	0.720	-	2.025
Our Model	Two split way	Case1 pRBFNN	3 (quadratic)		4/4	60	0.0083	-	0.0425
			3 (quadratic)		8/3	80	0.0139	-	0.0328
			3 3 2 3		4/4	50	0.0084	-	0.0265
			2 4 2 3 3 3 4 4		8/4	88	0.0046	-	0.0280
		Case2 pRBFNN	3 (quadratic)		6/4	90	0.0045	-	0.0294
			3 (quadratic)		8/3	80	0.0127	-	0.0294
			4 4 2 3 3 4		6/4	68	0.0057	-	0.0256
			1 1 2 2 3 3 3 2		8/5	83	0.0024	-	0.0216
	Three way split	Case1 pRBFNN	4 (M-quadratic)		8/5	168	0.0041	0.0438	0.0573
			3 (quadratic)		10/4	150	0.0067	0.0863	0.1481
			3 2 2 2 3 4 3 3		8/4	86	0.0064	0.0226	0.0266
			1 2 1 2 1 2 3 1 3 1		10/5	65	0.0121	0.0291	0.0615
		Case2 pRBFNN	3 (quadratic)		8/4	120	0.0037	0.0770	0.1007
			3 (quadratic)		10/4	150	0.0046	0.0767	0.0969
			2 3 2 2 4 2 3 1		8/4	62	0.0060	0.0187	0.0579
			1 1 2 3 2 2 3 2 2 2		10/4	62	0.0046	0.0334	0.0324

4.2 MPG Data

We consider the well-known automobile MPG data (http://archive.ics.uci.edu/ml/datasets/Auto+MPG) with the output being the automobile's fuel consumption

Table 2. Comparison of identification error with previous models

Model			Polynomial type of individual rules	No. of clusters /inputs	No. of parame-ters	PI	VPI	EPI
Regression model			2 (linear)	-	8	10.86	-	12.09
ANFIS[12]			2 (linear)	16/7	128	0.0851	-	541.32
Linguistic Model[13]	Without optimization		-	36/7	-	3.78		4.22
	One-loop optimization		-	36/7	-	2.90		3.17
	Multistep optimization		-	36/7	-	2.86		3.14
Our Model	Two split way	Case1 pRBFNN	4 (M-quadratic)	4/5	64	1.8926	-	3.0188
			4 (M-quadratic)	8/4	88	1.6661	-	3.0422
			2 3 4 1	4/5	47	2.0673	-	2.6844
			4 3 2 2 1 4 1 2	8/6	95	1.7929	-	2.5229
		Case2 pRBFNN	3 (quadratic)	4/5	84	1.7475	-	2.6434
			3 (quadratic)	6/4	90	1.6945	-	2.9336
			2 3 1 3	4/6	64	1.9366	-	2.5234
			3 2 1 1 2 3	6/7	90	1.8671	-	2.3974
	Three way split	Case1 pRBFNN	4 (M-quadratic)	4/5	64	1.9060	2.4412	3.7545
			3 (quadratic)	20/3	200	1.9944	2.5779	3.7168
			4 4 2 4	4/5	57	1.9887	2.2596	3.4351
			2 2 1 1 1 2 2 2 4 1 3 3 3 1 1 1 3 2 4 1	20/3	86	2.2771	2.4627	3.2462
		Case2 pRBFNN	4 (M-quadratic)	6/4	66	1.7941	2.5041	4.6333
			3 (quadratic)	30/2	180	2.3717	2.6684	3.3579
			2 4 2 2 2 2	6/6	57	1.8977	2.0149	3.6768
			2 2 1 1 1 2 2 2 4 1 3 3 3 1 1 1 3 2 4 1	30/2	91	2.2798	2.7048	3.3766

expressed in miles per gallon. Here we consider 392 input-out pairs. The number of input variable is 7. The number of input variables is varied from 1 to 6. The performance index is defined by (13).

The identification error (performance index) of the proposed model is also compared to the performance of some other models in Table 2.

5 Concluding Remarks

In this paper, we have proposed optimization methods and structure of polynomial radial basis function neural network. The pRBFNN is the extended architecture of the conventional RBFNN. The connection weight of proposed pRBFNN is represented as four kinds of polynomials, unlike in most conventional RBFNN constructed with constant as connection weight. Input space is partitioned with the aid of kernel functions and each kernel function is used as Gaussian type. Least Square Estimation (LSE) is used to estimate the coefficients of polynomial. Also, in order to design the pRBFNN model, the center value of each kernel function is determined based on HCM clustering algorithm. The width of RBF, the polynomial type of the consequent part of fuzzy rules and input variables are identified through PSO algorithm. To effectively evaluate the proposed model, the division of dataset is considered as two cases such as training and testing dataset or training, validation and testing dataset.

Acknowledgements. This work was supported by KESRI(2009T100100563), which is funded by MKE(Ministry Knowledge Economy) and also supported by the GRRC program of Gyeonggi province [GRRC SUWON2009-B2, Center for U-city Security & Surveillance Technology].

References

1. Frolov, A.A., Husek, D., Muraviev, I.P., Polyakov, P.Y.: A Boolean factor analysis by attractor neural network. IEEE Trans. Neural Netw. 3, 698–707 (2007)
2. Yuan, J.L., Fine, T.L.: Neural-network design for small training sets of high dimension. IEEE Trans. Neural Netw. 2, 266–280 (1998)
3. Park, J., Sandberg, I.: Universal approximation using radial-basis function networks. Neural Comput. 3, 246–257 (1991)
4. Chen, S., Billings, S.A.: Neural network for nonlinear dynamic system modeling and identification. Int. J. Control. 56, 319–346 (1992)
5. Xie, N., Leung, H.: Blind equalization using a predictive radial basis function neural network. IEEE Trans. Neural Netw. 16, 709–720 (2005)
6. Pedrycz, W.: Conditional Fuzzy Clustering in The Design of Radial Basis Function Neural Networks. IEEE Trans. Neural Network 9, 601–612 (1998)
7. Bezdek, J.C., Keller, J., Krisnapuram, R., Pal, N.R.: Fuzzy models and algorithms for pattern recognition and image processing. Kluwer Academic Publisher, Dordrecht (1999)
8. Tsekouras, G.E.: On the use of the weighted fuzzy c-means in fuzzy modeling. Advances in engineering Software 36, 287–300 (2005)
9. Kennedy, J., Eberhare, R.: Particle swarm optimization. In: IEEE Int. Conf. Neural Networks, pp. 1942–1948 (1995)

10. Oh, S.K., Pedrycz, W., Park, H.S.: Hybrid Identification in fuzzy-neural networks. Fuzzy Sets Syst. 138, 399–426 (2003)
11. Oh, S.K., Pedrycz, W., Park, H.S.: Rule-based multy-FNN Identification with the aid of evolutionary fuzzy granulation. J. Knowledge-Bases Syst. 17, 1–13 (2004)
12. Jang, J.R.: ANFIS: Adaptive-Network-Based Fuzzy Inference Systems. IEEE Trans. on Systems, Man and Cybernetics 23, 665–685 (1993)
13. Pedrycz, W., Kwak, K.C.: Linguistic Models as a Framework of User-Centric System Modeling. IEEE Trans. 36, 727–745 (2006)

Convergence of the Projection-Based Generalized Neural Network and the Application to Nonsmooth Optimization Problems

Jiao Liu, Yongqing Yang*, and Xianyun Xu

School of Science, Jiangnan University, Wuxi 214122, P.R. China

Abstract. This paper introduces a projection-based generalized neural network, which can be used to solve a class of nonsmooth convex optimization problems. It generalizes the existing projection neural networks for solving the optimization problems. In addition, the existence and convergence of the solution for the generalized neural networks are proved. Moreover, we discuss the application to nonsmooth convex optimization problems. And two illustrative examples are given to show the efficiency of the theoretical results.

Keywords: Nonsmooth optimization, Differential inclusions, Generalized neural network, Projection, Convergence.

1 Introduction

This paper is mainly concerned with the convergence and the application of the following generalized neural network

$$\begin{cases} \dot{z} \in -z + P_\Omega(z) - \alpha J(P_\Omega(z)) \\ x = P_\Omega(z) \end{cases} \tag{1}$$

where α is a positive constant, $z \in \mathbf{R}^n$, $J(\cdot)$ is a convex, closed-valued and locally bounded mapping from $\mathbf{R}^n$ to itself, Ω is a n-dimension convex closed subset of $\mathbf{R}^n$, $P_\Omega(\cdot)$ is a projection operator, which is defined as $P_\Omega(u) = arg\min_{v\in\Omega} \|u-v\|$, for $u \in \mathbf{R}^n$, and $x \in \mathbf{R}^n$ is the output solution trajectory of the generalized neural network. Meanwhile we call $z(t)$ the aided solution trajectory.

In recent years, many researchers have investigated the convergence and the application of some projection-based neural networks [1,2,3,4,5,6,7]. By means of this method, a plenty of nonlinear complementary problems, engineering problems, economical decision problems can be solved. Especially for the model $\frac{du}{dt} = \lambda(-\alpha G(P_\Omega(u)) + P_\Omega(u) - u)$ in [6], Xia obtained the asymptotical stability and the exponential stability without the smooth assumption of the nonlinear mapping. The neural network generalized the existing neural networks for

* This work was jointly supported by the National Natural Science Foundation of China under Grant 60875036, the Key Research Foundation of Science and Technology of the Ministry of Education of China under Grant 108067, and supported by Program for Innovative Research Team of Jiangnan University.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 254–261, 2010.

solving nonlinear optimization. In [7], Gao used the neural network to solve constrained variational inequality problems. However, they're merely suitable for some optimization problems whose objective functions and constraints are continuously differentiable. For some nonsmooth optimization problems, nonlinear mapping $G(\cdot)$ in [6] can be extended to be a set-valued one. In this way, a circuit is obtained whose dynamics is no longer described by some standard differential equations, but replaced with the differential inclusions [8].

With the need of solving such nonsmooth optimization problems, Forti et.al [9] proposed a programming circuit which generalized the model introduced by Kennedy and Chua in [10]. They turned to the account of a nonsmooth penalty approach to aim at solving a large class of nonsmooth optimization problems in real time. Since then, some generalized neural networks were proposed for solving optimization problems [11,12,13,14,15,16]. For example, Forti et.al [11] discussed the convergence of neural networks for programming problems via a nonsmooth Łojasiewicz inequality. Liu [13] proposed a one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming. In addition, the subgradient-based neural networks were proposed for nonsmooth convex and nonconvex optimization problems [15,16].

Motivated by the above discussions, in this paper, we introduce a projection-based generalized neural network. Based on the stability theory of differential inclusions, the convergence of the generalized neural network is proved. The application to nonsmooth optimization problems is also discussed.

This paper is organized as follows: Section 2 gathers some definitions and lemmas which will be used later in this paper. Section 3 is devoted to our main results on the analysis of existence and convergence of the solution. Section 4 discusses the application of the generalized neural network (1). After the numerical illustrations in Section 5, this paper concludes in Section 6.

2 Preliminaries

In this part, we will gather some toolkits which will be used in the remaining sections, including some definitions and lemmas. They are about projection operator, nonsmooth analysis and differential inclusions, which aims at paving a way to the analysis of convergence. And more details can be found in [16,17,18,19,20].

Lemma 1. *[17] Let Ω be a nonempty closed convex set in $\mathbf{R}^n$. For $\forall u, v \in \mathbf{R}^n$, the projection operator satisfies the following results:*

$$(i) \quad \|P_\Omega(u) - P_\Omega(v)\| \leqslant \|u - v\|$$

$$(ii) \quad \|P_\Omega(u) - w\|^2 \leq \|u - w\|^2 - \|P_\Omega(u) - u\|^2, \forall w \in \Omega$$

$$(iii) \quad (u - P_\Omega(u))^T (P_\Omega(u) - P_\Omega(v)) \geqslant 0$$

Definition 1. *[18](set-valued map and u.s.c.) Suppose that $E \subset \mathbf{R}^n$, for any $x \in E$, $x \multimap F(x)$ is said to be a set-valued map from E to $\mathbf{R}^n$ if the nonempty*

set $F(x) \subset \mathbf{R}^n$. A set-valued map $F : E \multimap \mathbf{R}^n$ with nonempty values is said to be upper semicontinuous at $x_0 \in E$ if for any open set N containing $F(x_0)$, there exists a neighborhood U of x_0 such that $F(x_0) \subset N$. F is upper semicontinuous on E iff its graph$\{(x, y) \in E \times \mathbf{R}^n : y \in F(x)\}$ is closed. And we denote upper semicontinuous as u.s.c. for short.

Definition 2. *[19](Clarke's generalized gradient of f(x)) The Clarke's generalized gradient of function $f : \mathbf{R}^n \to \mathbf{R}$ is defined as*

$$\partial f(x) = \{\xi \in \mathbf{R}^n : f^0(x; v) \geqslant < v, \xi >, \ for\ all \ \ v \in \mathbf{R}^n\},$$

where the generalized directional derivative $f^0(x; v)$ is

$$f^0(x; v) = \lim_{\substack{y \to 0 \\ t \to 0^+}} \frac{\sup\{f(y + tv) - f(y)\}}{t}.$$

Definition 3. *[19](regular) A function $f : \mathbf{R}^n \to \mathbf{R}$, which is locally Lipschitz near $x \in \mathbf{R}^n$, is said to be regular at x if $f^0(x; v) = f^{'}(x; v)$, where $f^{'}(x; v)$ is the usual one-sided directional derivative and equals to $\lim_{t \to 0} \frac{f(x+tv)-f(x)}{t}$. The function f is said to be regular in $\mathbf{R}^n$, if it is regular for any $x \in \mathbf{R}^n$.*

Remark 1. By virtue of [19], a locally Lipschitz and convex function in $\mathbf{R}^n$ is regular in $\mathbf{R}^n$. Moreover, if a function is continuously differentiable in $\mathbf{R}^n$, then it is regular in $\mathbf{R}^n$.

Lemma 2. *[19] Let f be Lipschitz near each point of an open convex subset U of x, then f is convex on U iff the multifunction ∂f is monotone on U; that is iff*

$$(x - x')^T(\xi - \xi^{'}) \geq 0, \forall x, x' \in U, \xi \in \partial f(x), \xi^{'} \in \partial f(x');$$

f is strongly convex, iff the multifunction ∂f is strongly monotone, i.e.

$$(x - x')^T(\xi - \xi^{'}) \geq \mu \|x - x'\|^T, \mu > 0, \ and\ \forall x, x' \in U, \xi \in \partial f(x), \xi^{'} \in \partial f(x'),$$

where ∂f is the Clarke's generalized gradient of $f(x)$.

Remark 2. Generally, the properties of monotone often play the important role in the analysis of the convergence, connecting with the convex function. We can see the relationship between these two aspects in Section 3.

Lemma 3. *[8] If $W : \mathbf{R}^n \to \mathbf{R}$ is regular at $x(t)$ and $x(\cdot) : \mathbf{R} \multimap \mathbf{R}^n$ is differentiable at t and locally Lipschitz continuous near t, then*

$$\frac{d}{dt} W(x(t)) = < \xi, \dot{x}(t) >, \forall \xi \in \partial W(x(t)).$$

Lemma 4. *[16] Let $x(t)$ be a global solution to the generalized neural network (1). Suppose that there exists $V : \mathbf{R}^n \to \mathbf{R}$ such that $V(x(t))$ is absolutely continuous on $[t_0, +\infty)$, and there exists $\epsilon > 0$ such that for almost all t for which $x(t) \in \{x : V(x) > 0\}$, we have*

$$\frac{d}{dt}V(x(t)) \leq -\epsilon.$$

Then, the trajectory $x(t)$ hits $\{x : V(x) \leq 0\}$ in finite time and stays there thereafter.

Definition 4. *If the aided solution z^* of model (1) satisfies that $0 \in -\bar{z} - \alpha\partial f(P_\Omega(z^*)) + P_\Omega(z^*)$, then z^* is said to be the equilibrium point of model (1), as well as the output solution trajectory.*

Next, we will pay more attention to studying the convergence of the equilibrium point of the generalized neural network (1).

3 Convergence Analysis

Now, let's turn to the topic of the main results of this paper. First of all, we'll give some assumptions for convenience.

Assumptions:

1. The set-valued map $J : \mathbf{R}^n \multimap \mathbf{R}^n$ is nonempty and u.s.c. over $z \in \mathbf{R}^n$.
2. There exists one equilibrium point $z^* \in \mathbf{R}^n$.

Theorem 1. *Let the generalized neural network (1) satisfy Assumptions 1 and 2. Then there exists at least one aided solution trajectory $z(t)$ of model (1) for any given initial point $z(t_0)$ over $[t_0, +\infty)$.*

Proof. Let $T(z) = z + \alpha J(P_\Omega(z))$. Just in light of Assumption 1 and the conditions in model (1), we can obtain that $T(z)$ is nonempty, convex, closed-valued, u.s.c. and locally bounded over $z \in \mathbf{R}^n$. Meanwhile, from the definition of $P_\Omega(z)$ and Lemma 1, we know that $P_\Omega(z)$ is nonempty, convex, closed-valued, u.s.c. and locally bounded over $z \in \mathbf{R}^n$ as well. Considering all of the analysis above, there exists at least one absolutely continuous aided solution trajectory $z(t)$ for model (1) for any given initial point $z(t_0)$ over $[t_0, T)$, according to [21].

Moreover, we will show that T can be extended to $+\infty$. Indeed, considering that there exists $\gamma \in J(P_\Omega(z))$ such that $\dot{z} = -z + P_\Omega(z) - \alpha\gamma$, we have $\dot{z} + z = P_\Omega(z) - \alpha\gamma$. Hence, we can integrate it from t_0 to $t(> t_0)$ and get the result:

$$\begin{aligned} z(t) &= e^{-(t-t_0)}z(t_0) + e^{-t}\int_{t_0}^{t} e^s(P_\Omega(z(s)) - \alpha\gamma(s))ds \\ &= e^{-(t-t_0)}z(t_0) + e^{-t}\int_{t_0}^{t} e^s P_\Omega(z(s))ds + e^{-t}\int_{t_0}^{t} e^s(-\alpha\gamma(s))ds \end{aligned}$$

Obviously, by the mean-valued integrable theorem and the properties of the set-valued map $J(P_\Omega(z))$, we have

$$\begin{aligned}\|z(t)-z^*\| &= \|e^{-(t-t_0)}z(t_0)+e^{-t}\int_{t_0}^{t}e^s P_\Omega(z(s))ds\\ &\quad +e^{-t}\int_{t_0}^{t}e^s(-\alpha\gamma(s))ds-z^*\|\\ &\le e^{-(t-t_0)}\|z(t_0)\|+e^{-t}\|P_\Omega(\hat{z})\|\int_{t_0}^{t}e^s ds\\ &\quad +e^{-t}\int_{t_0}^{t}e^s(\|-\alpha\gamma(s))\|+\|z^*\|)ds\\ &\le (1-e^{-(t-t_0)})(\|P_\Omega(\hat{z})\|+\alpha M)+e^{-(t-t_0)}(\|z(t_0)\|+\|z^*\|)\end{aligned}$$

where $\|\gamma\| \le M$, by virtue of the locally boundness of $J(P_\Omega(z))$. Therefore, $\|z(t)-z^*\| \le N = \max\{\|P_\Omega(\hat{z})\|+\alpha M, \|z(t_0)\|+\|z^*\|\}$. In other words, $z(t)$ is bounded with the center z^* and the radius N for any $t > t_0$. Hence, T can be extended to $+\infty$. Thus the proof is completed.

Then, the next theorem will show that the output solution trajectory of the generalized neural network (1) can be convergent within the finite time.

Theorem 2. *Let the generalized neural network (1) satisfy Assumption 1, 2. If the set-valued map J is strongly monotone with the parameter $\mu > 0$, then the output solution trajectory $x(t) = P_\Omega(z)$ is convergent to $x^* = P_\Omega(z^*)$ within the finite time and stays there thereafter for a.a. t.*

Proof. Here, we'll prove this theorem step by step.

1. Let $V(z) = \frac{1}{2}\|z-P_\Omega(z^*)\|^2 - \frac{1}{2}\|z-P_\Omega(z)\|^2$, where z^* is the equilibrium point of the generalized neural network (1).

   Actually, in light of (ii) of Lemma 1, by choosing that $w = P_\Omega(z^*) \in \Omega, u = z$, we can obtain that

   $$\|P_\Omega(z)-P_\Omega(z^*)\|^2 \le \|z-P_\Omega(z^*)\|^2 - \|z-P_\Omega(z)\|^2.$$

   Obviously, $V(z) \ge \frac{1}{2}\|P_\Omega(z)-P_\Omega(z^*)\|^2 \ge 0$.
2. Calculate the derivative of $V(z)$.

   By means of that $0 \in -z^* - \alpha\partial f(P_\Omega(z^*)) + P_\Omega(z^*)$, i.e. there exists $\gamma^* \in J(P_\Omega(z^*))$ such that $0 = -z^* - \alpha\gamma^* + P_\Omega(z^*)$, then we can get that

$$\begin{aligned}\tfrac{dV}{dt} &= [(z-P_\Omega(z^*))-(z-P_\Omega(z))]\cdot\dot{z}\\ &\le \sup_J[(P_\Omega(z)-P_\Omega(z^*))^T(-z+P_\Omega(z)-\alpha J(P_\Omega(z)))]\\ &= \sup_J[(P_\Omega(z)-P_\Omega(z^*))^T(-z+P_\Omega(z)-\alpha J(P_\Omega(z))+z^*+\\ &\quad \alpha\partial f(P_\Omega(z^*))-P_\Omega(z^*))]\\ &= \sup_J[-(P_\Omega(z)-P_\Omega(z^*))^T(z-z^*)+\|P_\Omega(z)-P_\Omega(z^*))\|^2\\ &\quad -\alpha(P_\Omega(z)-P_\Omega(z^*))^T\cdot(J(P_\Omega(z))-J(P_\Omega(z^*)))]\\ &\le \sup_J[-\|P_\Omega(z)-P_\Omega(z^*)\|^2+\|P_\Omega(z)-P_\Omega(z^*)\|^2-\alpha(P_\Omega(z)-P_\Omega(z^*))^T\\ &\quad \cdot(J(P_\Omega(z))-J(P_\Omega(z^*)))]\\ &\le \sup_J[-\alpha(P_\Omega(z)-P_\Omega(z^*))^T(J(P_\Omega(z))-J(P_\Omega(z^*)))]\end{aligned}$$

3. Noting that J is strongly monotone with the parameter μ, we can get

$$(P_\Omega(z) - P_\Omega(z^*))^T(J(P_\Omega(z)) - J(P_\Omega(z^*))) \geq \mu\|P_\Omega(z) - P_\Omega(z^*)\|^2,$$

where $\mu > 0$. So,

$$\frac{dV}{dt} \leq -\alpha\mu\|P_\Omega(z) - P_\Omega(z^*)\|^2 \leq 0.$$

As a result, it shows that $V(z)$ is decreasing and we can easily get that the output solution is globally asymptotically stable at the point $P_\Omega(z^*)$. On the other hand, if $\|P_\Omega(z) - P_\Omega(z^*)\|^2 = 0$ within some finite time t_0, then $P_\Omega(z) - P_\Omega(z^*) = 0$. It shows that $V(z) = 0$ at that time. By virtue of inequality (2), we know the decreasing monotone of the function $V(z)$ and it shows that $V(z)$ will stay at zero over $[t_0, +\infty)$. Hence, $\frac{dV}{dt} = 0$ and then $\|P_\Omega(z) - P_\Omega(z^*)\|^2 = 0$. This tells us that the output solution converges at the point $P_\Omega(z^*)$ in finite time.

If not, we have that $\|P_\Omega(z) - P_\Omega(z^*)\|^2 > \epsilon > 0$ forever. So $\frac{dV}{dt} < -\alpha\mu\epsilon < 0$. According to Lemma 4, the theorem also can be proved.

If J is monotone, we can easily get the following corollary through the proof of Theorem 2.

Corollary 1. *Let the generalized neural network (1) satisfy Assumption 1, 2 and J be monotone, then the output solution trajectory $x(t) = P_\Omega(z)$ is globally asymptotically stable at $x^* = P_\Omega(z^*)$.*

4 Application

Generally, the smooth convex optimization can be dealt with by using the differential quotient of the objective functions. However, when the object function $f(x)$ is nonsmooth, ∇f can not be used again. But the generalized gradient of $f(x)$ will do.

Next, we will Consider the following nonsmooth optimization problem:

$$\begin{cases} \min f(x) \\ s.t. \quad x \in \Omega \end{cases} \tag{2}$$

where Ω is a nonempty closed convex subset of $\mathbf{R}^n$ and $f(x)$ is nonsmooth convex and Lipschitz near each point of an open convex subset of $\mathbf{R}^n$. Then we surely can build the generalized neural network (3) by replacing the set-valued map J with $\partial f(\cdot)$.

$$\begin{cases} \dot{z} \in -z + P_\Omega(z) - \alpha\partial f(P_\Omega(z)) \\ x = P_\Omega(z) \end{cases} \tag{3}$$

Remark 3. If f is convex on U, according to Lemma 2, we can obtain that $\partial f(P_\Omega(z))$ is monotone at $P_\Omega(z)$, that is $(P_\Omega(z) - P_\Omega(z'))^T(\xi - \xi^{'}) \geq 0, \forall z, z' \in U, \xi \in \partial f(P_\Omega(z)), \xi^{'} \in \partial f(P_\Omega(z'))$;

If f is strongly convex, we have $(P_\Omega(z) - P_\Omega(z'))^T(\xi - \xi^{'}) \geq \mu\|P_\Omega(z) - P_\Omega(z')\|^2, \forall z, z' \in U, \xi \in \partial f(P_\Omega(z)), \xi^{'} \in \partial f(P_\Omega(z'))$, where $\mu > 0$.

Corollary 2. *Let the object function of problem (2) be Lipschitz and convex, then there exists at least one absolutely continuous aided solution trajectory $z(t)$ of the model (3) over $[t_0, +\infty)$. Moreover, the output solution trajectory $x(t)$ is globally asymptotically stable at $x^*(t) = P_\Omega(z^*)$ over $[t_0, +\infty)$ for a.a. t.*

Corollary 3. *Let the object function of problem (2) be Lipschitz and strongly convex, then the output solution trajectory $x(t)$ of the model (3) converges to $x^*(t) = P_\Omega(z^*)$ over $[t_0, +\infty)$ within the finite time for a.a. t.*

5 Numerical Examples

In this section, to illustrate the effectiveness of results obtained in the previous section, we consider the following nonsmooth optimization problems.

Example 1

$$\begin{aligned} &\min f(x) = |x_1| + |x_2| + |x_3| \\ &\textit{subject to} \quad \Omega = \{x \in \mathbf{R}^3 : x_1^2 + x_2^2 + x_3^2 \leq 1\} \end{aligned}$$

where the object function $f(x)$ is convex. Obviously, the optimal solution is $x^* = (0, 0, 0)^T$ and $f(x^*) = 0$. We use the model (3) to solve it. In this example, the projection operator $P(\cdot)$ is defined as

$$P_\Omega(x_i) = \begin{cases} x_i, & x_1^2 + x_2^2 + x_3^2 \leq 1 \\ x_i / \sqrt{x_1^2 + x_2^2 + x_3^2}, & x_1^2 + x_2^2 + x_3^2 > 1. \end{cases}$$

We choose step size $\Delta t = 0.01, \alpha = 2$. For 20 random initial points, simulation results show the optimal solution is obtained. Fig. 1 displays the convergence behavior of the generalized neural network (3).

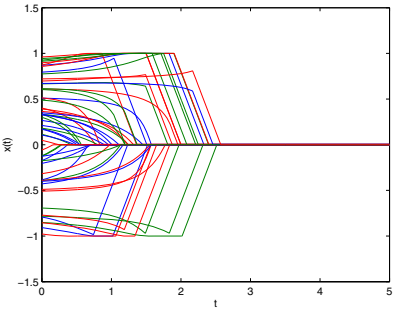


Fig. 1. The trajectories of Example 1 with 20 random initial values

6 Conclusion

This paper mainly investigates the convergence of a class of projection-based generalized neural network and the applications to the nonsmooth convex

optimization problems. The existence and the convergence of the solution in the finite time are discussed via the Lyapunov method and the stability theory of the differential inclusions. Two numerical examples are provided to show the efficiency of the results.

References

1. Liu, Q., Cao, J., Xia, Y.: A delayed neural network for solving linear projection equations and its analysis. IEEE Trans. Neu. Net. 16, 834–843 (2005)
2. Xia, Y., Feng, G.: On convergence conditions of an extended projection neural network. Neu. Comp. 16, 515–525 (2005)
3. Ma, R., Chen, T.: Recurrent neural network model based on projective operator and its application to optimization problems. Appl. Math. Mech. 27(4), 543–554 (2006)
4. Yang, Y., Cao, J.: A delayed neural network method for solving convex optimization problems. IJNS 16(4), 295–303 (2006)
5. Hu, X.: Applications of the general projection neural network in solving extended linear-quadratic programming problems with linear constraits. Neu. Comp. 72, 1131–1137 (2009)
6. Xia, Y., Feng, G.: A new neural network for solving nonlinear projection equations. Neu. Net. 20, 577–589 (2007)
7. Gao, X., Liao, L.: A new projection-based neural network for constrained variational inequalities. IEEE Trans. Neu. Net. 20(3), 373–388 (2009)
8. Forti, M., Nistri, P., Quincampoix, M.: Generalized neural nerwork for nonsmooth nonlinear programming problems. IEEE Trans. Cir. 51(9), 1741–1754 (2004)
9. Forti, M., Nistri, P.: Global convergence of neural networks with discontinuous neuron activations. IEEE Trans. Cir. 50(11), 1421–1435 (2003)
10. Kennedy, M.P., Chua, L.: Neural network for nonlinear programming. IEEE Trans. Cir. 35(5), 554–562 (1988)
11. Forti, M., Nistri, P., Quincampoix, M.: Convergence of neural networks for programming problems via a nonsmooth Łojasiewicz inequality. IEEE Trans. Neu. Net. 17(6), 1471–1486 (2006)
12. Liu, Q., Wang, J.: A one-layer recurrent neural network with a discontinuous activation function for linear programming. Neu. Comp. 20, 1366–1383 (2008)
13. Liu, Q., Wang, J.: A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming. IEEE Trans. Neu. Net. 19(4), 558–570 (2008)
14. Cheng, L., Hou, Z., Tan, M., Wang, X., Zhao, Z., Hu, S.: A recurrent neural network for non-smooth nonlinear programming problems. In: IJCNN 2007, pp. 12–17 (2007)
15. Xue, X., Bian, W.: Subgradient-based neural networks for nonsmooth convex optimization problems. IEEE Trans. Cir. 55(8), 2378–2391 (2008)
16. Bian, W., Xue, X.: Subgradient-based neural networks for nonsmooth nonconvex optimization problems. IEEE Trans. Neu. Net. 20(6), 1024–1038 (2009)
17. Noor, M.A., Wang, Y., Xiu, N.: Some new projection methods for variational inequalities. Appl. Mathe. Comput. 137, 423–435 (2003)
18. Lu, W., Wang, J.: Convergence analysis of a class of nonsmooth gradient systems. IEEE Trans. Cir. 55(11), 3514–3527 (2008)
19. Clarke, F.H.: Optimization, nonsmooth analysis. John Wiley Sons, Chichester (1983)
20. Aubin, J.P., Cellina, A.: Differential inclusions. Springer, Heidelberg (1984)
21. Aubin, J.P.: Viability Theory. Birhäuser, Cambridge (1991)

Two-Dimensional Adaptive Growing CMAC Network

Ming-Feng Yeh

Department of Electrical Engineering, Lunghwa University of Science and Technology, Taoyuan, 33327 Taiwan
mfyeh@mail.lhu.edu.tw

Abstract. This study attempts to develop a two-dimensional (2D) adaptive growing cerebellar model articulation controller network, which is constructed by connecting several 1D CMACs as a two-level tree structure. Without requiring the knowledge of the target function in advance, the number of states for each 1D CMAC as well as the number of CMACs is gradually increased during the adaptive growing process. Then the input space can be adaptively quantized by the proposed adaptive growing mechanism. In addition, the linear interpolation scheme is applied to calculate the network output and for simultaneously improving the learning performance and the generalization ability. Simulation results show that the proposed network not only has the adaptive quantization ability, but also can achieve a better learning accuracy with less memory requirement. Besides, the proposed network also could perform the best generalization ability among all considered models and, in general, attain a faster convergence speed.

Keywords: Cerebellar model articulation controller, Tree structure, Adaptive quantization, Pseudo-inverse, Linear interpolation.

1 Introduction

Cerebellar model articulation controller (CMAC) network, proposed by Albus [1], [2], is a kind of supervised neural network inspired by the human cerebellum. This network learns input-output mappings based on the premise that similar inputs should produce similar outputs. Therefore, unlike multi-layer feedforward networks, CMAC networks store information locally. Owing to its fast learning speed, good generalization ability and ease of implementation by hardware, the CMAC network has been successfully applied in many applications such as control problem, signal processing and pattern recognition [3]-[5].

Albus' CMAC network has two major limitations: enormous memory requirement for solving high-dimensional problem and difficult in selecting the memory structure parameters [6], [7]. Several studies had proposed to solve one of those two limitations, such as hierarchical CMAC (HCMAC) network [6], tree-type CMAC structure [8], macro structure CMAC (MS_CMAC) network [9], and so on. This study attempts to propose a two-dimensional (2D) adaptive growing cerebellar model articulation controller (AG-CMAC) network which is constructed by connecting several 1D CMACs as a two-level tree structure. The term "adaptive growing" means that the input space can be adaptively quantized by the proposed growth mechanism. There

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 262–271, 2010.

are two growth processes in this study. The first one is the state growth process which is used to adaptively quantize the domain of one direction, say *x*-axis, by gradually inserting some new knots into the specific regions. Inserting new knots will cause some increase in the number of states (i.e., state growth). Note that the state growth process is analogous to the previous work [10]. The second one is the CMAC growth process which is used to adaptively quantize the domain of the other direction, say *y*-axis, by gradually increasing the number of 1D CMACs in a certain region. That is to say, not only the number of the 1D CMACs but also the number of the states is gradually increased during the adaptive growing process. With the help of those two growth processes, the proposed AG-CMAC network can simultaneously achieve the purpose of self-organizing input space and overcome two major limitations of Albus's CMAC network. Note that this study only focuses on the 2D problem because the multidimensional problems can be solved by use of the hierarchical structure as the HCMAC network [6].

2 Conventional CMAC Network

In a CMAC network, each state variable is quantized and the problem space is divided into discrete states [1], [2]. A vector of quantized input values specifies a discrete state and is used to generate addresses for retrieving information from memory elements for this state. Fig. 1 illustrates a simple block division of 2D CMAC network. This simple example has two state variables, x_1 and x_2, with each quantized into three discrete regions, called blocks, in each layer. Areas, such as *Bb*, *Ee*, and *Ih*, formed by quantized regions are called hypercubes. Only the blocks on the same layer can be combined to form a hypercube. The CMAC associates each hypercube to a physical memory address. Information for a discrete state is distributively stored in memory elements associated with the hypercubes being addressed.

Assume that there are N_s quantized states in the input space and that information for a quantized state is distributively stored in N_e memory elements. Let N_{nem} represent the entire memory size ($N_{nem} > N_e$). Then the stored data y_k (the actual output of the CMAC) for the state s_k is the sum of stored contents of all addressed hypercubes and can be expressed as

$$y_k = \mathbf{a}_k \mathbf{w} = \sum_{l=1}^{N_{nem}} a_{k,l} w_l, \tag{1}$$

where w_l, $l = 1, 2, 3, \ldots, N_{nem}$, is the content of the lth memory element, $\mathbf{w} = [w_1, w_2, \ldots, w_{N_{nem}}]^T$, $a_{k,l}$ is the association index indicating whether the lth memory element is addressed by the state s_k, and $\mathbf{a}_k = [a_{k,1}, a_{k,2}, ..., a_{k,N_{nem}}]$. Since each state addresses exactly N_e memory elements (hypercubes), only those addressed $a_{k,l}$'s are 1, and the others are 0. The CMAC uses a supervised learning method to adjust the memory contents during each learning cycle. Its updating rule can be described as

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \frac{\eta}{N_e} \mathbf{a}_k^T [\hat{y}_k - \mathbf{a}_k \mathbf{w}(t)], \quad t = 1, 2, 3, ... \tag{2}$$

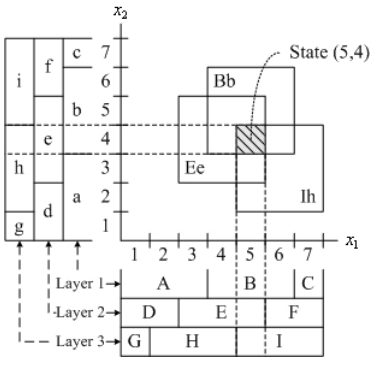


Fig. 1. Block division of 2D CMAC network

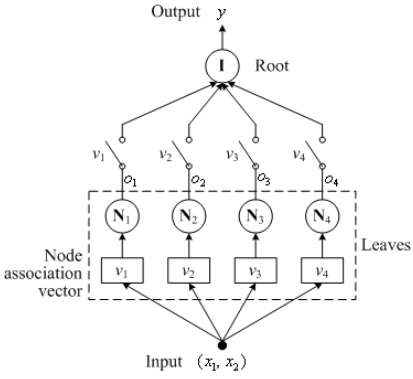


Fig. 2. Structure of 2D AG-CMAC network

where $\mathbf{w}(t+1)$ is the memory content at time $t+1$, $\mathbf{w}(t)$ is the one at previous time t, η is the learning rate, $\hat{y}_k$ is the desired value for the state s_k, and $\hat{y}_k - \mathbf{a}_k \mathbf{w}(t)$ is the error for the state s_k. Note that only the addressed memory elements are updated. The analytical solution of $\mathbf{w}$ is

$$\mathbf{w}^* = \mathbf{A}^+ \hat{\mathbf{y}}, \tag{3}$$

where $\hat{\mathbf{y}}^T = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_{N_s}]$ and $\mathbf{A}^+ = \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1}$ is the pseudo-inverse of the association matrix $\mathbf{A}$ formed from the association vector $\mathbf{a}_k$, k = 1, 2, 3, …, N_s, as row vector. That is to say, the updating rule (2) could converge to $\mathbf{w}^*$ if the learning rate is properly selected [11].

The block division of 1D CMAC network is very similar to Fig. 1, but just takes a single variable into consideration. Besides, the 1D CMAC associates each block, not the hypercube, to a physical memory address. Information for a state is stored in the memory elements associated with those blocks being addressed.

3 Two-Dimensional Adaptive Growing CMAC Network

The proposed 2D AG-CMAC network is constructed by connecting several 1D CMACs as a two-level tree structure as given in Fig. 2. Although the proposed structure is similar to the two-level tree structure of MS_CMAC network [9], three main differences between them could be briefly stated as follows. The first one is that the root node in the MS_CMAC network is a 1D CMAC, but the last layer of the proposed network is an interpolation node. The second one is that all 1D CMACs (nodes) in the 2D AG-CMAC network could be dynamically updated in utilizing, but the leaf nodes in the MS_CMAC network are static. Other nodes in the MS_CMAC network are still dynamical. The last one is that the 2D AG-CMAC network allows both the number of states and the number of 1D CMACs to be gradually increased during the learning process, but the MS_CMAC can not.

3.1 Quantization of 2D Input Space

Assume that the proposed 2D AG-CMAC network is constructed by M 1D CMACs, the ith CMAC, denoted by N_i, $i = 1, 2, \ldots, M$, is quantized into $N_{s(i)}$ states by the knots $q_{i,j}$'s, $j = 0, 1, 2, \ldots, N_{s(i)}$, and the considered 2D input space is $S = [x_1^L, x_1^U] \times [x_2^L, x_2^U]$, where the superscripts L and U represent the lower and upper bounds of the input variable x_1 or x_2, respectively. Then those 1D CMACs could partition the 2D input space into several subspaces and there are $\sum_{i=1}^{M} N_{s(i)}$ states in the input space S. Besides, without lost of generality, the active parameter for each 1D CMAC in the network hereafter is assumed to be x_2. This means that the output of any 1D CMAC is only depended on the value of x_2. Fig. 3 demonstrates a possible quantization of the input space for the case of $M = 4$. The subspace $[g_{i-1}, g_i] \times [x_2^L, x_2^U] := R_i$ represents the region covered by the ith CMAC, where the knots on the x_1-axis, g_i's, are used to partition the domain $[x_1^L, x_1^U]$. Note that the region R_i is not the input space of the ith CMAC. In fact, the input space of any 1D CMAC in this example is $[x_2^L, x_2^U]$. The area formed by $[g_{i-1}, g_i] \times [q_{i,j-1}, q_{i,j}] := U_i \times V_{i,j}$ represents the quantization region associated with the state $s_{i,j}$ (the jth state in the ith CMAC). For example, the gray cell in Fig. 3 represents the region covered by the state $s_{3,2}$.

3.2 Network Operations

When an arbitrary input pair, say $\mathbf{p} = (x_1', x_2')$, is applied to the proposed network, some temporary pairs generated from the input pair are used to determine which CMAC is active. Assume that $\mathbf{p} \in s_{i,j}$ (the jth state of the ith CMAC). The relation between the temporary pairs and the given input pair is depicted in Fig. 4. In general, an input pair will generate two temporary pairs: one is $\mathbf{t}_i = (\bar{x}_{1,i}, x_2') \in s_{i,j}$ and the other is $\mathbf{t}_{i-1} = (\bar{x}_{1,i-1}, x_2') \in s_{i-1,k}$. These two temporary pairs, located in adjacent states but in different CMACs, could indicate which CMACs are active. In this case there exist two active CMACs, i.e., $v_i = v_{i-1} = 1$. However, when $x_1' < \bar{x}_{1,1}$ or $x_1' > \bar{x}_{1,M}$, there is only one temporary pair, $(\bar{x}_{1,1}, x_2')$ or $(\bar{x}_{1,M}, x_2')$, to be generated. At this situation, only one CMAC is active. To sum up, determining which CMACs are active can be simply represented by

$$\begin{cases} \text{If } x_1' < \bar{x}_{1,1}, \text{ then } v_1 = 1. \\ \text{If } \bar{x}_{1,i-1} \le x_1' \le \bar{x}_{1,i},\ i = 2, 3, ..., M, \text{ then } v_i = v_{i-1} = 1. \\ \text{If } x_1' > \bar{x}_{1,M}, \text{ then } v_M = 1. \end{cases} \tag{4}$$

Note that those CMACs which are not identified as active are inactive.

Once the node association vector $\mathbf{v} = [v_1, v_2, \ldots, v_M]$ is determined, the output of each leaf node can be obtained by

$$o_i(x_1', x_2') = v_i(x_1')\mathbf{N}_i(x_2'),\ i = 1, 2, \ldots, M, \tag{5}$$

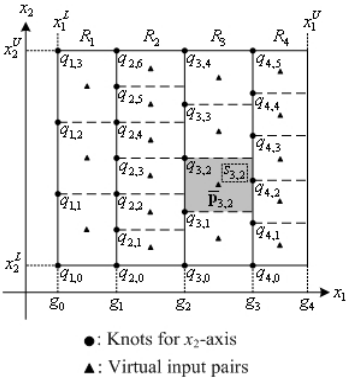


Fig. 3. Input space quantization of 2D AG-CMAC network

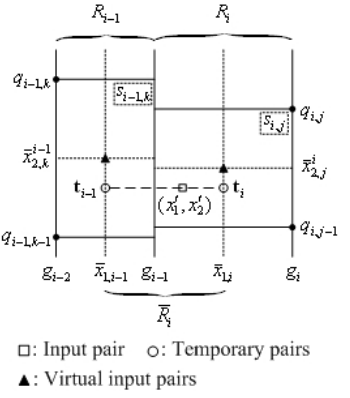


Fig. 4. Relation between the temporary pairs and the input pair

where $v_i(x_1')$ denotes the ith node association index due to x_1' and $\mathbf{N}_i(x_2')$ represents the output of the ith CMAC due to x_2'. The switches between the root and leaf layers are also controlled by the node association vector. If $v_i = 1$, the corresponding switch is closed; otherwise, it is opened. A leaf node can deliver its output to the root node if the corresponding switch is closed. Finally, the root node performs the linear interpolation on its receiving values to obtain the network output as follows.

$$y(x_1', x_2') = \begin{cases} o_1, & \text{if } x_1' < \bar{x}_{1,1}, \\ \dfrac{\bar{x}_{1,i} - x_1'}{\bar{x}_{1,i} - \bar{x}_{1,i-1}} o_{i-1} + \dfrac{x_1' - \bar{x}_{1,i-1}}{\bar{x}_{1,i} - \bar{x}_{1,i-1}} o_i, & \text{if } \bar{x}_{1,i-1} \le x_1' \le \bar{x}_{1,i}, \\ o_M, & \text{if } x_1' > \bar{x}_{1,M}, \end{cases} \tag{6}$$

where $1 < i \le M$.

3.3 Virtual Training and Testing Patterns

Let the target function be in the form of $y = f(x_1, x_2)$ and $\mathbf{p}_{\alpha,\beta} = (x_{1,\alpha}, x_{2,\beta})$ be a given input pair in S, where $\alpha = 1, 2, \ldots, N_{p1}$ and $\beta = 1, 2, \ldots, N_{p2}$. By this definition, there are $N_{p1} \times N_{p2} := N_p$ given pairs in S. Denote $N(U_i)$ as the number of the input points $x_{1,\alpha}$'s in U_i, $N(V_{i,j})$ as the number of the input points $x_{2,\beta}$'s in $V_{i,j}$, and $N(s_{i,j})$ as the number of input pairs belonging to the state $s_{i,j}$. Then it can be shown that $N(s_{i,j}) = N(U_i) \times N(V_{i,j})$.

Two kinds of virtual patterns, state training patterns the state testing patterns, also generated from the given patterns are used in the state growth process. The state training patterns are in the form of $(\bar{\mathbf{p}}_{i,j}, \bar{d}_{i,j})$, where $\bar{\mathbf{p}}_{i,j}$ and $\bar{d}_{i,j}$ are defined by

$$\bar{\mathbf{p}}_{i,j} = \frac{1}{N(s_{i,j})} \sum_{\mathbf{p}_{\alpha,\beta} \in s_{i,j}} \mathbf{p}_{\alpha,\beta}, \tag{7}$$

$$\bar{d}_{i,j} = \frac{1}{N(s_{i,j})} \sum_{\mathbf{p}_{\alpha,\beta} \in s_{i,j}} d_{\alpha,\beta}. \tag{8}$$

It is obvious that there are exactly $\sum_{i=1}^{M} N_{s(i)}$ state training patterns and no two distinct training patterns are associated with the same state. The state training patterns allow the proposed network to be trained by a few instances. In Fig. 3, the black triangle in each cell represents the corresponding virtual input pair.

To compute the state errors, every CMAC output must be known in advance. This will result in a heavy computational complexity during the entire state growth process. In order to reduce the computational complexity on calculating the state errors, each CMAC is provided with a set of testing patterns. The state testing patterns for the *i*th CMAC are in the form of $(\tilde{\mathbf{p}}_{i,\beta}, \tilde{d}_{i,\beta})$, where the input pair $\tilde{\mathbf{p}}_{i,\beta} = (\bar{x}_{1,i}, x_{2,\beta})$ and the corresponding desired output $\tilde{d}_{i,\beta}$ are defined as follows.

$$(\bar{x}_{1,i}, x_{2,\beta}) = \frac{1}{N(U_i)} \sum_{x_{1,\alpha} \in U_i} (x_{1,\alpha}, x_{2,\beta}), \tag{9}$$

$$\tilde{d}_{i,\beta} = \frac{1}{N(U_i)} \sum_{x_{1,\alpha} \in U_i} f(x_{1,\alpha}, x_{2,\beta}). \tag{10}$$

By this way, there are only $N(V_{i,j})$ input pairs $\tilde{\mathbf{p}}_{i,\beta}$'s belonging to the state $s_{i,j}$. That is, the number of testing patterns for the state $s_{i,j}$ is reduced from $N(U_i) \times N(V_{i,j})$ to $N(V_{i,j})$.

3.4 State Growth Process

Adaptive growing process of the proposed network involves two main stages. One is the state growth stage. The other is the CMAC growth stage. Note that the latter always follows the former during the proposed adaptive growing process. Learning starts with the network composed of a small number of 1D CMACs, e.g., $M = 4$ in Fig. 2. Those 1D CMACs could quantize the problem space into several discrete states as shown in Fig. 3. Once the 2D input space is quantized, the state training patterns could be determined by (7) and (8). Then letting $\hat{y}_j = \bar{d}_{i,j}$, $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N_{s(i)}$, enables the memory contents of the *i*th CMAC to be attained by the pseudo-inverse scheme (3). That is the so-called one-step training method for any CMAC in the network.

Once every CMAC has been trained, the RMSE for the state $s_{i,j}$ due to the state testing patterns can be obtained by

$$e_s(s_{i,j}) = \sqrt{\frac{1}{N(V_{i,j})} \sum_{x_{2,\beta} \in V_{i,j}} [\tilde{d}_{i,\beta} - o_i(\bar{x}_{1,\alpha}, x_{2,\beta})]^2}. \tag{11}$$

Those states with larger state error must be further quantized to minimize the generalization caused by quantization. The evaluation criterion is that a state, say $s_{i,j}$, needs to be repartitioned if $e_s(s_{i,j}) \geq \rho$, where ρ is a predefined threshold. If the state $s_{i,j}$ is evaluated to be repartitioned, it can be achieved by inserting the new knot $\bar{x}_{2,j}^{i}$ (the x_2 element of $\bar{\mathbf{p}}_{i,j}$) into the associated interval $(q_{i,j-1}, q_{i,j}]$. If there are more than one state

which must be repartitioned at the same time, each associated interval must be inserted a corresponding new knot for re-quantization. The state growth process for each CMAC will stop when a specified performance measure or a pre-specified maximum number of states (or epochs) is reached.

Once any state is repartitioned, the state training patterns must be regenerated by (7) and (8) according to the newly quantized space. Each CMAC has its own state growth process. Different CMACs therefore may have different quantization results as illustrated in Fig. 3. Since the state growth process will not affect the values of the state testing patterns $(\tilde{\mathbf{p}}, \tilde{d})$'s, they need not update in this stage.

3.5 CMAC Growth Process

Once the state growth stage is completed, each given input pair $\mathbf{p}_{\alpha,\beta}$ is applied to the proposed network such that the network outputs could be used to evaluate where need to insert some new CMACs for further improving the learning accuracy. Let $\bar{x}_{1,0} = x_1^L$ and $\bar{x}_{1,M+1} = x_1^U$, and define the ith interpolation region as $\bar{R}_i = [\bar{x}_{1,i-1}, \bar{x}_{1,i}] \times [x_2^L, x_2^U]$, $i = 1, 2, \ldots, M+1$ as given in Fig. 4. Then the region $\bar{R}_i$, $i \neq 1$ and $M+1$, are associated with the $(i-1)$th and ith CMACs. Note that the outmost regions, $\bar{R}_1$ and $\bar{R}_{M+1}$, is covered by only a single CMAC, i.e., the first and the last one, respectively. The RMSE for the kth interpolation region, termed the region error, is defined as

$$e_R(\bar{R}_k) = \sqrt{\frac{1}{N(\bar{R}_k)} \sum_{\mathbf{p}_{\alpha,\beta} \in \bar{R}_k} [d_{\alpha,\beta} - y(x_{1,\alpha}, x_{2,\beta})]^2}\ , \tag{12}$$

where $N(\bar{R}_k)$ represents the number of input pairs in the region $\bar{R}_k$. Then the $(k-1)$th and kth CMACs need to be repartitioned if $e_R(\bar{R}_k) \geq \delta$, where δ is a predefined repartition threshold for the CMAC growth process. The repartition interval is defined as the interval covered the $(k-1)$th and kth CMACs, i.e., $[g_{k-2}, g_k]$. It is possible that there are r, $r \geq 1$, successive regions $\bar{R}_j$'s, $j = k, k+1, ..., k+r-1$, satisfied that $e_R(\bar{R}_j) \geq \delta$. At this situation, the repartition interval is defined as $[g_{k-2}, g_{k+r-1}]$ which is covered by $(r+1)$ CMACs. Since there are r interior points in the repartition interval $[g_{k-2}, g_{k+r-1}]$, repartitioning that interval can be simply implemented by linearly spacing one or two more interior points between the knots g_{k-2} and g_{k+r-1}. The CMAC growth process will stop when a specified performance measure or a pre-specified maximum number of CMACs (or epochs) is reached.

3.6 Adaptive Growing Algorithm

The algorithm for the overall adaptive growing process is presented below.

1) Start with a network composed of a small number of 1D CMACs.
2) Uniformly quantize the input space of each CMAC into several states.
3) Determine whether the stopping criterion of CMAC growth process is fulfilled or not. If fulfill, terminate the process; otherwise, go to Step 4.

4) Perform state growth algorithm as follows.
   4.1) Produce the state testing patterns by (9) and (10).
   4.2) Generate the state training patterns by (7) and (8).
   4.3) Obtain the memory contents for each CMAC by (3).
   4.4) Calculate the 1D CMAC output o_i for each state training pattern by (5).
   4.5) Determine whether the stopping criterion of state growth process is fulfilled. If fulfill, terminate the state growth process; otherwise, go to Step 4.6.
   4.6) Compute all state errors $e_s(s_{i,j})$'s by (11).
   4.7) Insert a new knot into the state with state error larger than the repartition threshold ρ, and then return to Step 4.2. If there are more than one state which need to be repartitioned simultaneously, each state must be assigned a corresponding new knot.
5) Perform the following CMAC growth algorithm and then return to Step 2.
   5.1) Apply each given pattern **p** into the network and then calculate the network output y by (6).
   5.2) Compute all region errors by (12).
   5.3) Determine all possible repartition intervals satisfied the evaluation criterion for CMAC growth.
   5.4) Quantize each repartition interval by linearly spacing one or two more interior points.

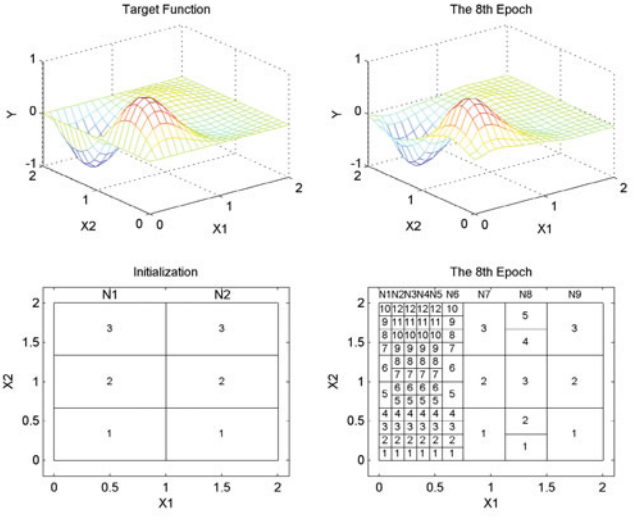


Fig. 5. Simulation results

4 Simulation Results

Consider a 2D AG-CMAC network trained to generate the target function $f(x_1, x_2) = 2\exp(-2x_1)\sin(\pi x_1)\sin(\pi x_2)$, where $(x_1, x_2) \in [0.0, 2.0]\times[0.0, 2.0]$. The network initially is constructed by only two 1D CMACs with each corresponding space being equally quantized into three states as shown in the bottom left of Fig. 5. Each state is

distributively stored in three memory elements, i.e., $N_e = 3$. Besides, the repartition thresholds for both the state growth stage and the CMAC growth stage are set to be 0.05, i.e., $\rho = \delta = 0.05$. For each 1D CMAC, the pre-specified maximum number of learning cycles for the state growth process is 3.

While the network is trained by 441 (21^2) training instances which are uniformly distributed in the input space, the learning result and the corresponding quantization of the input space at the end of the 8th epoch are also depicted in Fig. 5. Since the target function in the region of [0.75, 2.00]×[0.0, 2.0] is very close to a flat surface, the proposed approach actually quantizes that region into a small amount of states by use of three 1D CMACs. The remaining region is partitioned into 68 states by use of other six CMACs. In this example, the testing set contains 961 (41^2) instances where around 25% of instances have been trained. Table 1 compares the results of 2D AG-CMAC network with other CMAC models. In the table, "1–N CMAC" represents that the number of nodes in the root and leaf layers for MS_CMAC network are 1 and N, respectively, and "N_1×N_2 CMAC" represents that the quantization numbers of the x_1- and x_2-axes for Albus' CMAC are N_1 and N_2, respectively. Those comparison models are also trained with the learning rate η of 0.1 and the generalization size N_e of 3.

In this example, the ratio of computational time for each Albus' CMAC becomes around 0.60, but two MS_CMAC models require much more computational time than the proposed network. Generally speaking, the proposed network can attain better results in the training and testing accuracies, except that the 21×21 CMAC outperforms the 2D AG-CMAC network in the learning accuracy. Since each training instance is corresponding to one and only one state in the 21×21 CMAC, this manner could effectively reduce the learning interference during the learning process. Hence the 21×21 CMAC can achieve the best training result. However, its generalization ability still remains poorer than the proposed network's.

Table 1. Comparison of AG-CMAC with other CMACs

Model	Number of States	Memory Size	RMSE		Computational Time Ratio
			Training	Test	
AG-CMAC	79	97	0.0497	0.0415	1.0000
MS_CMAC					
1-9 CMAC	81	110	0.1026	0.1097	1.0578
1-12 CMAC	144	182	0.0632	0.0701	1.2053
Albus' CMAC					
9×12 CMAC	108	52	0.1163	0.1070	0.5920
12×12 CMAC	144	66	0.0904	0.0876	0.5920
15×15 CMAC	225	97	0.0779	0.0668	0.5920
18×18 CMAC	324	134	0.0729	0.0627	0.6267
21×21 CMAC	441	177	0.0205	0.0539	0.6267

5 Conclusions

This study developed a 2D AG-CMAC network which is constructed by several CMACs as a two-level tree structure. In the proposed network, not only the number of

states for each CMAC but also the number of CMACs is gradually increased during the adaptive growing process. Without requiring the knowledge of the target function in advance and with the help of the proposed adaptive growing mechanism, the input space can be adaptively quantized during the learning process. In addition, the proposed network also utilizes the linear interpolation scheme to smooth the network output. Both the adaptive growing mechanism and the linear interpolation scheme enable the 2D AG-CMAC network to achieve a better learning performance with less memory requirement. Besides, the proposed network also could perform the best generalization ability among all considered models and, in general, attain a faster convergence speed.

Acknowledgments. This work was supported by the National Science Council, Taiwan, Republic of China, under Grants NSC 97-2221-E-262-009 and NSC 98-2221-E-262-004.

References

1. Albus, J.S.: A new approach to manipulator control: The cerebellar model articulation controller (CMAC). J. Dyn. Syst. Meas. Contr. Trans. ASME 97(3), 220–227 (1975)
2. Albus, J.S.: Data storage in the cerebellar model articulation controller (CMAC). J. Dyn. Syst. Meas. Contr. Trans. ASME 97(3), 228–233 (1975)
3. Yeh, M.F.: Single-input CMAC control system. Neurocomputing 70, 2638–2644 (2007)
4. Glanz, F.H., Miller, W.T., Kraft, L.G.: An overview of the CMAC neural network. In: Proc. 1991 IEEE Neural Netw. Ocean Eng., pp. 301–308. IEEE Press, New York (1991)
5. Tao, T., Lu, H.C., Hung, T.H.: The CA-CMAC for downsampling image data size in the compressive domain. In: Proc. 2002 IEEE Int. Conf. Syst. Man Cybern., vol. 5. IEEE Press, New York (2002)
6. Lee, H.M., Chen, C.M., Lu, Y.F.: A self-organizing HCMAC neural-network classifier. IEEE Trans. Neural Netw. 14(1), 15–27 (2003)
7. Menozzi, A., Chow, M.Y.: On the training of a multi-resolution CMAC neural network. In: Proc. IEEE Int. Symp. Ind. Electron., vol. 3, pp. 1201–1205. IEEE Press, New York (1997)
8. Lane, S.H., Handelman, D.A., Gelfand, J.J.: Theory and development of higher-order CMAC neural networks. IEEE Contr. Syst. Mag. 12, 23–30 (1992)
9. Hung, S.L., Jan, J.C.: MS_CMAC neural-network learning model in structural engineering. J. Computing Civil Engrg., ASCE 13(1), 1–11 (1999)
10. Yeh, M.F., Chang, K.C.: Adaptive growing quantization for 1D CMAC network. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5551, pp. 118–127. Springer, Heidelberg (2009)
11. Horváth, G., Szabó, T.: Kernel CMAC with improved capability. IEEE Trans. Syst. Man Cybern. B 37(1), 124–138 (2007)

A Global Inferior-Elimination Thermodynamics Selection Strategy for Evolutionary Algorithm

Fahong Yu, Yuanxiang Li, and Weiqin Ying

State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China
fhyu520@126.com

Abstract. In this paper, a new evolutionary algorithm based on global inferior-elimination thermodynamics selection strategy (IETEA) is proposed. Also, a definition of the two-dimensional entropy (2D entropy) of the particle system is given and the law of entropy increment is applied to control the algorithm running. The purpose of the new algorithm is to systematically harmonize the conflict between selective pressure and population diversity while searching for the optimal solutions. The new algorithm conforms to the principle of minimal free energy in simulating the competitive mechanism between energy and entropy in annealing process. By solving some typical high-dimension problems with multiple local optimizations, satisfactory results are achieved. The results show that this algorithm has preferable capability to avoid the premature convergence effectively and reduce the cost in search to some extent.

Keywords: Evolutionary algorithm, Free energy, Entropy, Selection strategy.

1 Introduction

The evolutionary algorithm is wildly used in the fields of searching, optimization, and machine learning [1], in which connotative parallel character and efficiency of using global information are two prominent characteristics. However, it often suffers from the premature convergence because of the loss of population diversity at the early stage and the extremely slow search especially in some high-dimension problems with multiple local optimizations.

To harmonize the contradiction between exploration and exploitation in evolutionary algorithm, a wide variety of improvements in evolutionary algorithms were raised such as scaling the fitness, sharing the fitness, and driving all individuals moving as the most possible [2] etc. But they have been not yet sufficiently systematical and effective for some optimization problems. Mori and his fellows have proposed the thermodynamics genetic algorithm (TDGA) [6] combined SA and GA. Although TDGA was effective to solve some problems, it was unsatisfied performance with extremely high cost in computation and could not be directly applied to solve numerical optimization problems with real-coded since the calculation of entropy was closely related with the encoding method. To improve the stability and decrease the computational cost of TDGA, a global inferior-elimination thermodynamics selection strategy for evolutionary algorithm (IETEA) is proposed in this paper.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 272–279, 2010.

2 Brief Thermodynamic Background on IETEA

The main idea of thermodynamic evolutionary algorithm (TEA) was an introduction of competition mechanism between energy and entropy in the process of solid annealing to evolutionary algorithm. In an annealing process, a metal with high temperature and disorder is regarded as a system which is slowly cooled to the approximate thermodynamics equilibrium at any temperature [6]. As the cooling, the system will become to be in order gradually and approach a "frozen" state at the temperature coming to zero.

The second law of thermodynamics has described an isolated system is always evolving towards the direction of increasing entropy and its formula can be described in algebra as the express $dS \geq 0$, which means that the isolated system will evolve towards the stable status with the continually increasing of the entropy.

3 Description of IETEA

The population and the individuals in GA may be regarded as a thermodynamics system and particles. Then the energy, the entropy and the temperature is served as the fitness, the measurement of population diversity and the controllable weight parameter respectively. Every population state is exactly full of global optimization that can be interpreted as the ground state. This analogy provides an approach for IETEA to simulate the competitive mechanism between energy and entropy in annealing to systematically harmonize the conflicts between selective pressure and population diversity in GA.

3.1 2D Entropy and Measurement of Population Diversity

It is a critical part how to measure population diversity when introducing the competitive mechanism into GA. Therefore, in order to measure population diversity more effectively, 2D entropy defined in search space is introduced.

Definition 1: Let $P=\{X_1,X_2,\ldots,X_n\} \subseteq H'$ be a population in the search space H', the variable X_i be a individual of the population P, $X_i = (x_{i1},x_{i2},\ldots,x_{im})^T$, $X_i \in H' \cap H' \subset R^m$, $i=1,2,\ldots,n$, the variable d_i indicate the distance between X_i and the center of search space H', the variable $R=max\{d_i|i=1,2,\ldots,n\}$ be the largest radius and the variable R be equally divided into several segments $d=R/r$, $(u-1)d \leq R < ud$, $u=1,2,\ldots r$, the method of dividing is called the dividing in distance and if the express $d_i \in R_u$ is true, the individual X_i is regarded the individual belonged to the radius scope R_u. Here $\sum_{j=1}^{r} R_j = R \cap 1 \leq j < r+1 \cap j \in N \cap (R_1=R_2=\ldots=R_r)$.

Definition 2: Let a vector X_d be a individual of the population $P=\{X_1,X_2,\ldots, X_n\} \subseteq H'$ and $d_d=R$, and the value $\cos\theta_{id} = \left(\sum_{k=1}^{m} x_{ik}x_{dk}\right) / \left[\sum_{k=1}^{m} x_{ik}^2 \sum_{k=1}^{m} x_{dk}^2\right]^{1/2}$ is the cosine of the angle between the vector X_i and the vector X_d, if the field $[min(\cos\theta_{id}), max(\cos\theta_{id})]$

is divided into continual fields C_v, $(C_v=l)\cap(v=1,2,\ldots c)\cap(v-1)l\leq C_v < vl$, then the method of dividing is called the dividing in angle and if $\cos\theta_{id} \in C_v$, the individual X_i is regarded the individual belonged to the angle scope C_v.

Definition 3: Let $P=\{X_1,X_2,\ldots,X_n\} \subseteq H^{'}$ be a population in the search space $H^{'}$, $X_i = (x_{i1},x_{i2},\ldots,x_{im})^T$, $X_i \in H^{'}\cap H^{'} \subset R^m$, $i=1,2,\ldots,n$, C_v and R_u is the representation of the v-th angle scope and the radius scope u-th respectively, $v=1, 2,\ldots c$, $u=1,2,\ldots r$, if the individual $X_i \in C_v\cap X_i \in R_u$, then the common field is called two-dimensional field (2D field) Q_j. here $j=1,2,\ldots,k$, $k=rc$. Let the function $g(X_i, Q_j)$ be a index function, $\pi=\{Q_j|1\leq j\leq k\cap j\in N\}$, if the express $X_i \in C_v\cap X_i \in R_u$ is true, then the equation $g(X_i, Q_j)=1$ be true, otherwise, the equation $g(X_i, Q_j)=0$ be true. The count of individuals in population P can be calculated as the formula $n_j = \sum_{i=1}^{n} g(X_i, Q_j)$, $X_i \in P$ and the express $S(P)$ is called two-dimensional entropy (2D entropy) of population P. Here the entropy is denoted as:

$$S(P) = -\sum_{j=1}^{k} \frac{n_j}{n} \log_k \frac{n_j}{n} . \tag{1}$$

According to 2D entropy, the measurement of diversity of population is in a sphere with the best solution as the center and utmost distance from the best solution to a particle among the current population as radius. If the number of the particles belonged to Q_j is $n_j(i=1, 2,\ldots, k)$, the probability about the particle X_i belonged to radius scope R_j is the value $p_i = n_j/n$. Here the conditions accord with the follow expresses $1\leq j\leq k+1\cap j\in N$, $\sum_{i=1}^{k} n_j = n$, $\sum_{j=1}^{k} p_j = 1$. So the value of 2D entropy can be calculated according to probability distributed in every two-dimensional field and be applied to measure the diversity of population efficiently instead of depending on the mode of particle's encoding. In the course of calculation of 2D entropy, the division of every 2D field manifests a principle that more near the center of the space 2D field is, smaller field the 2D field is, which justly guaranteed the diversity of current population.

3.2 Minimization of Free Energy at Each Temperature

In order to apply the principle of minimal free energy in thermodynamics to IETEA, the method of free energy of population is presented as follow.

Definition 4: For $P_t=\{X_1, X_2, \ldots, X_n\} \in H^n$, $E(P_t)$ is called the population energy of population P_t where $E(P_t) = \frac{1}{n}\sum_{i=1}^{n} e(X_i)$.

Here the energy $e(X_i)$ is the value $f(X_i)$ if the optimized problem is minimum $min(f(X))$ or the energy $e(X_i)$ is the value $-f(X_i)$ if the optimized problem is maximum $max(f(X))$.

Definition 5: For $P_t=\{X_1,X_2,\ldots,X_n\} \in H^n$, $\pi=\{R_j|1\leq j<k+1\cap j\in N\}$, $F(\pi, T, P_t)$ is called the free energy of population P_t at temperature T for partition π where

$$F(\pi,T,P_t)=E(P_t)-TS(\pi,P_t)\ . \tag{2}$$

The aim of algorithm is to minimize the free energy of population at each temperature T_t during the process of evolution so as to drive the thermodynamic system towards equilibrium state.

3.3 Global Inferior-Elimination Thermodynamic Selection Strategy

As an idea strategy, its task is to generate next generation population P_{t+1} from population P_t and sub-population O_t and to sure the free energy of population P_{t+1} is minimal. However, the time-complexity to minimize the free energy of each generation population precisely was up to $O(nC_{n+m}{}^{n})$, which was unfeasible in practical execution. A greedy selection of thermodynamics strategy (greedy thermodynamic selection, GTS) was presented in TDGA, in which the time-complexity to minimize the free energy of each generation is $O(mn^2)$ and the effectiveness could not be guaranteed. At beginning of every population P_{t+1}, the selection pressure always keeps maximization since the entropy is zero and with the proceeding of the selection, the selection pressure decreased slowly. It was clear that the GTS could not reflect the competition between energy and entropy completely. So a global inferior-elimination thermodynamic selection strategy (ETS) was presented in this paper.

For a population $P_t^q = \{X_1, X_2, \ldots, X_q\} \in H^q$, $1\le q\cap q\in N$, if $(j-1)R_t \le d_{i,opt} < jR_t \cap 1\le j<q\cap j\in N$, then $g(X_i, R_{tj})=1$, otherwise $g(X_i, R_{tj})=0$. $n_j=\sum_{i=1}^{q} g(X_i,R_{tj})$, $\sum_{i=1}^{k} n_i = q$, $X_i \in P_t^q$. A temporary variable $\overline{S}(\pi, P_t^{q-1})$ representing possible entropy of the population P_t^{q-1} where a particle has been eliminated can be calculated according to follow formula.

$$\overline{S}(P_t^{q-1}) = -\sum_{i=1}^{k}\frac{n_i}{q-1}\log_k\frac{n_i}{q-1} = \frac{q}{q-1}\left(-\sum_{i=1}^{k}\frac{n_i}{q}\log_k\frac{n_i}{q}-\sum_{i=1}^{k}\frac{n_i}{q}\log_k\frac{q}{q-1}\right)$$

$$= \frac{q}{q-1}\left(S(P_t^q)-\log_k\frac{q}{q-1}\sum_{i=1}^{k}\frac{n_i}{q}\right) = \frac{q}{q-1}\left(S(P_t^q)-\log_k\frac{q}{q-1}\right)$$

Similarly, another temporary variable $\overline{E}(P_t^{q-1})$ representing possible entropy of the population P_t^{q-1} where a particle has been eliminated can be calculated according to follow formula.

$$\overline{E}(P_t^{q-1}) = \frac{q}{q-1}E(P_t^q)\ . \tag{3}$$

So the free energy $F(\pi, T, P_t^{n-q}, X_u)$ of population $P_t^{q-1} - \{X_u\}$ at temperature T for partition π can be got according to following formulation.

$$F(\pi, T, P_t^{q-1}, X_u) = E(P_t^{q-1}) - TS(P_t^{q-1})$$

$$= \frac{q}{q-1}F(\pi,T,P_t^q) - \frac{1}{q-1}e(X_u) - \frac{1}{q-1}T\left(n_{r(u)}\log n_{r(u)} - (n_{r(u)}-1)\log(n_{r(u)}-1) - \left(q\log q-(q-1)\log(q-1)\right)\right)$$

Here the variable X_u is a representation of eliminated particle and the variable $n_{r(u)}$ is a counting number to denote how many particles of the population belong to the 2D field where the particle X_u belong to.

ETS (π, T, P_t , O_t) is described as follows:

Step 1: Produce an interim population P_t^{n+m} by appending the offspring population O_t to the parent population P_t^n, divide 2D field according to *definition 2* and *definition 3*, let *selected[m+n]=1, q=1,* and get count the number of particles in each 2D field and the value of free energy $F(\pi, T, P_t^{n+m})$.

Step 2: Set $i=0$, $F_{min}= F(\pi, T, P_t^{n+m})$.
Step 3: If *selected[i]==0, then i++*. if $i <= m+n$, then go to Step 3, else go to Step 6.
Step 4: If $F(\pi, T, P_t^{n+m-q}, X_i) < F_{min}$ then $F_{min} = F(\pi, T, P_t^{n+m-q}, X_i)$, $no=i$.
Step 5: Set i++. If $i < =m+n$, then go to Step 3.
Step 6: Set $F(\pi, T, P_t^{n+m-q}) = F_{min}$, *selected[no]=0, q++*, if $q<=m$, go to Step 2.
Step 7: Form next generation population P_{t+1} constituted by particles in population P_t^{n+m} with corresponding value of the array selected is nonzero.

It is clear that the competition between the energy and entropy do not carry on in nowhere during whole process of ETS and the time complexity is lowed to $O((n+m)m)$.

3.4 Procedure of IETEA

Procedure of IETEA is described as follows:

Step 1: Create particles randomly with size n as an initial population P_0, determine the value of *k, m,* T_0 *and MAXGENS* and set $t=0$;
Step 2: Generate offspring population O_t with m particles by uniform crossover and mutation.
Step 3: $P_{t+1}= ETS(m+n)$.
Step 4: t++, $T_t=T_0 /(1+t/100)$.
Step 5: If *Termination_test*(P_t)==*False*, then go to Step 2.
Step 6: Output the final results.

4 Experiments and Results Analysis

The significance of a new optimization strategy depends on the effectiveness of solving practical problems. Generally, it is more complicated for some high-dimension problems with multiple local optimizations or some optimization problems required more precise solution. In this section, three typical high-dimension problems with multiple local optimizations are solved by IETEA and gave the compared results with other methods so as to test the performance and reliability of each optimization algorithm. Here three functions are selected from test set of Benchmark and denoted with notations f_1 and f_2 respectively. The dimensions of each function are set at 30.

Test 1: $\min f_1(x) = \sum_{i=0}^{n-1} (x_i^2 - 10\cos(2\pi x_i) + 10)$, $-5.12 \le x_i \le 5.12$

Test 2: $\min f_2(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$, $-30 \le x_i \le 30$

The first function f_1, called rastrigin function, is a typical high-dimension global minimization problems with multiple local minimization and the global minimization is $minf_1(x)=0$ when $X=(0, 0, \ldots, 0)^T$. There are about 10n local minimization in its search space $[1.25, 1.25]^n$.

The second function f_2, called Ackley function, is an optimized problems with multiple local minimization and the global minimization is $minf_2(x)=0$ when $X=(0, 0, \ldots, 0)^T$.

In this experiment, four algorithms included Guotao algorithm (GTGA) [8], particle swarm optimization (PSO), thermodynamics genetic algorithm (TDGA) and IETEA are applied. All of them are utilized uniform crossover operator with the probability $P_c=0.8$, the uniform mutation operator with the probability $P_m=0.05$, and the same population size $n=30$.The termination condition is satisfied when the minimal fitness of the function is less than 1×10^{-20} or the generation is more than *MAXGENS*. The size *m* of offspring population for TDGA and IETEA is *10* respectively. The four algorithms are carried out on functions f_1 and f_2 on the same machine and gets the mean results, the worst results and the best results respectively for 30 times. The compare of statistic results are given as table 1.

Table 1. Experiment results for test 1, test2 and test3

Algorithm		**Best**	**Mean**	**Worst**
Test 1	GTGA	4.2563E-2	1.5600E+2	2.0664E+2
	TDGA	2.5764E-2	1.0047E+1	2.2001E+1
	PSO	3.2042 E-4	1.2901E+2	5.8513 E+3
	IETEA	1E-20	1E-20	1E-20
Test 2	GTGA	8.3051E-3	7.04506	8.9713
	TDGA	2.4274 E-3	2.72963	4.2742
	PSO	2.4989 E-3	1.0394E-1	1.4969E+1
	IETEA	1E-20	1E-20	1E-20

Through the compare of every instance of the mean results, the worst results and the best results performed with GTGA, TDGA, PSO and IETEA, it is clear that the result carried out in IETEA is the best than other algorithms. The convergence curves for test 1 and test2 on GTGA, TDGA, PSO and IETEA were presented as fig.1 and fig2 respectively.

Viewing from fig.1 and fig2 some conclusion can be drawn that IETEA could obtain very good accuracy, stability and faster rate to converge comparing other three algorithms for its strong search capabilities in every generation, reliability to maintain the right search direction and ability to avoid premature convergence effectively. So it is regarded as a very important algorithm for some engineering optimization problems.

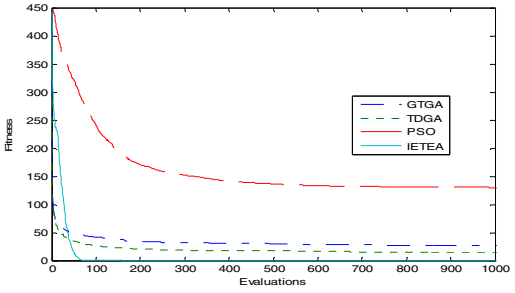


Fig. 1. Convergence curves of function f_1 on GTGA, TDGA, PSO and IETEA

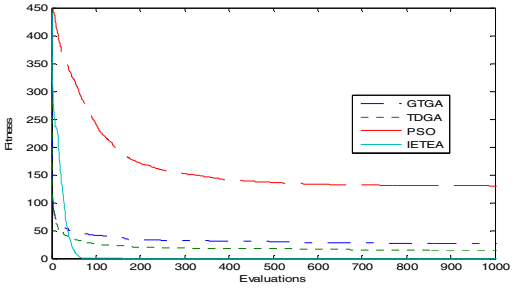


Fig. 2. Convergence curves of function f_2 on GTGA, TDGA, PSO and IETEA

The main reason why the outstanding performance could be archived for IETEA lies in its adaptive capability to maintain the dynamic balance between "selective pressure" and "species diversity" by adjusting the temperature in the process of ETS. Generally, in competition between "selective pressure" and "species diversity", the entropy occupies the predominance for high temperature in the early stages of evolution which lows level of aggregation and energy plays main function for low temperature in the latter evolution which allows a higher degree of aggregation contributing to converge to the optimization solution set.

5 Conclusion

In this paper, attempting to systematically harmonize the conflict between selective pressure and population diversity while searching for the optimal solutions, an improvement of evolutionary algorithm based on global inferior-elimination thermodynamics selection strategy is proposed. In IETEA, the definition of 2D entropy to measure the diversity of population and the introduction of global inferior-elimination thermodynamic selection strategy are given. The new algorithm reduces the complexity of the algorithm to some extent. By solving some typical testing problems, the efficiency and good performance of the IETEA were tested. Therefore, IETEA avoids premature convergence successfully and obtains faster rate, good accuracy and stability.

Further research will concentrate on more examinations on the other problems such as TSP and the analysis of the convergence traits about IETEA from the viewpoint of statistical mechanism.

Acknowledgment. This work was supported by the National Natural Science Foundation of China under Grant No. 60773009, the National High-Tech Research and Development Plan of China under Grant No.2007AA01Z290, and the Natural Science Foundation of Hubei Province of China under Grant No.2007ABA009.

References

1. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
2. Whitley, D.: The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In: Schaffer, J. (ed.) Proceedings of the 3rd International Conference on Genetic Algorithms, pp. 116–123. Morgan Kaufmann, Los Altos (1989)
3. Fukuda, T., Mori, K., Tsukiyama, M.: Parallel search for multi-modal function optimization with diversity and learning of immune algorithm. In: Dipankar, D. (ed.) Artificial Immune Systems and Their Applications, pp. 210–219. Springer, New York (1999)
4. Hu, T., Li, Y.X., Ding, W.: A new dynamical evolutionary algorithm based on the principle of minimal free energy. In: Kang, L.S., Cai, Z.H., Yan, X.S. (eds.) Progress in Intelligence Computation and Applications, pp. 749–754. China University of Geosciences, Wuhan (2005)
5. Kita, H., et al.: Multi-objective Optimization by means of Thermodynamic Genetic Algorithm. In: Proceedings of Parallel Problem Solving from Nature IV (PPSN-IV), pp. 504–512 (1996)
6. Mori, N., Yoshida, J., Tamaki, H., Kita, H., Nishikawa, Y.: A thermodynamic selection rule for the genetic algorithm. In: Fogel, D.B. (ed.) Proc. of IEEE Conf. on Evolutionary Computation, pp. 188–192. IEEE Press, New Jersey (1995)
7. Mori, N., Kita, H.: The entropy evaluation method for the thermodynamic selection rule. In: Banzhaf, W., Daida, J., Eiben, A.E., Garzon, M.H., Honavar, V., Jakiela, M., Smith, R.E. (eds.) Proc. of the Genetic and Evolutionary Computation Conference, p. 799. Morgan Kaufmann Publishers, San Francisco (1999)
8. Guo, T., Kang, L.S.: A new Evolutionary Algorithm for Function Optimization. Wuhan University Journal of Nature Science 4, 409–414 (1999)

Particle Swarm Optimization Based Learning Method for Process Neural Networks

Kun Liu, Ying Tan*, and Xingui He

Key Laboratory of Machine Perception, Ministry of Education,
Peking University, Beijing 100871, China
ytan@pku.edu.cn
http://www.cil.pku.edu.cn

Abstract. This paper proposes a new learning method for process neural networks (PNNs) based on the Gaussian mixture functions and particle swarm optimization (PSO), called PSO-LM. First, the weight functions of the PNNs are specified as the generalized Gaussian mixture functions (GGMFs). Second, a PSO algorithm is used to optimize the parameters, such as the order of GGMFs, the number of hidden neurons, the coefficients, means and variances of Gaussian functions and the thresholds in PNNs. In PSO-LM, the parameter space is transformed from the function space to real number space by using GGMFs. PSO can give a global search in the real number parameter space by avoiding the premature and gradient calculations in back propagation method. According to our analysis and several experiments, PSO-LM can outperform current basis function expansion based learning method (BFE-LM) for PNNs and the classic back propagation neural networks (BPNNs).

Keywords: Process Neural Networks, Learning Method, Particle Swarm Optimization, Gaussian Mixture Model.

1 Introduction

Recently, He et al. proposed this process neural networks (PNNs) to deal with process inputs, which can be related to time, locations etc. [1]. Inputs such as time series are usually sensitive to the time parameter, so the weights and thresholds should also be related to time in order to accumulate the effects of inputs more precisely. In PNNs, there exist process neurons (PNs). The inputs, connection weights and threshold of a PN can all be functions of time. Theoretically, the approximation capabilities of PNNs are better than classic artificial neural networks (ANNs) when solving for time-varying function problems. ANNs are considered as the special case of PNNs. PNNs have been applied in many actual tasks, such as simulation of oil reservoir exploitation [1] and traffic flow prediction [2].

In this study, we propose this new learning method for PNNs. Instead of transforming the inputs and weight functions, we specify the weight functions connected to the process neurons as the generalized Gaussian mixture functions (GGMFs), which theoretically can approximate any continuous functions. Afterward, instead of using the

* Corresponding Author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 280–287, 2010.

classic back propagation method, we use PSO to optimize the parameters, such as the order and coefficients of GGMFs, the means and variances of Gaussian functions, the number of hidden neurons and their thresholds. This new learning method for PNNs is called PSO-LM for short.

The remainder of this paper is organized as follows. Section 2 describes the original PNN model and a standard PSO model. The PSO-LM is proposed in Section 3 with discussions about its qualities. Section 4 elaborates two experiments to compare the PSO-LM with the existing BFE-LM for PNNs and the BPNN. Concluding remarks are drawn in Section 5.

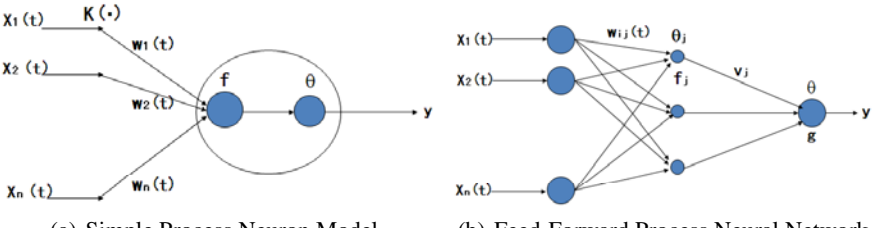
2 Background

2.1 Process Neural Networks

PNN is a generalized model of traditional neural networks, the inputs are related to instantaneous conditions [1]. The key of PNNs is the special structure of PNs. Fig. 1(a) shows a simple process neuron model (SPN), in which $x_i(t)(i = 1, 2, \ldots, n)$ are the input time-varying functions and $w_i(t)(i = 1, 2, \ldots, n)$ are the corresponding weight functions. $K(\cdot)$ is the aggregate function of the SPN, which can be summation or integration etc.. $f(\cdot)$ is the activation function, with a threshold value θ. The output of the SPN is a constant. A simple feed forward process neural network (PNN) is shown in Fig. 1(b).

The current learning methods for PNNs include: basis function expansion method (BFE) [1] and numerical learning method (NL) [3]. The BFE includes two steps: first, transform the input data and the weight functions of the PNN into the space that expanded by some specified orthogonal basis functions. Therefore, there will be no time-varying parameters in the model. Second, adjust the parameters in the networks by back propagation method. However, how to choose appropriate basis functions and how many expansion items should be reserved are awkward to decide. Furthermore, transforming the input data in the first place certainly will lose information contained in the input signals, which can restrain the performance of PNNs.

The NL method is only suitable for small-scale discrete input signals. When the length of the input series increases, the number of parameters in NL will increase



(a) Simple Process Neuron Model (b) Feed-Forward Process Neural Network

Fig. 1. Process Neural Networks

rapidly, which can lead to huge computational complexity. Furthermore, these two learning methods both rely on the back propagation (BP) method, which will suffer from the premature convergence problem and gradient calculations.

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) [4], as a stochastic global optimization technique based on a social interaction metaphor, can be used to train ANNs effectively [5].

Bratton and Kennedy defined a standard PSO model (SPSO) in [6]. The update rules can be equivalently transformed and expressed as follows:

$$\begin{aligned} V_{id}(t+1) &= \omega V_{id}(t) + c_1\epsilon_1(P_{iBd}(t) - X_{id}(t)) \\ &\quad + c_2\epsilon_2(P_{nBd}(t) - X_{id}(t)), \end{aligned} \tag{1}$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1). \tag{2}$$

where $i = 1, 2, \cdots, n$, n is the number of particles in the swarm, $d = 1, 2, \cdots, D$, and D is the dimensionality of solution space. The learning factors c_1 and c_2 are nonnegative constants, r_1 and r_2 are random numbers uniformly drawn from the interval $[0, 1]$, which are all scalar quantities for each particle in each dimension. P_{iBd} and P_{nBd} are the locations of the best positions found so far by particle i and its neighbors in dimension d. ω is called inertia weight, which determines the effect of the inertia to the movements of particles.

3 The Proposed PSO-LM for PNNs

It is impossible to learn the weight functions directly from the function space. In this study, we assume that all weight functions are GGMFs based on the fact that all the continuous functions can be approximated by GGMFs. Afterwards, the constant parameters in the model are optimized by PSO.

3.1 Generalized Gaussian Mixture Functions

A standard Gaussian mixture model is defined in (3), and its qualities are expressed in Theorem 1. For the proof of the theorem, one can refer to [7].

$$f = \sum_{i=1}^{k} a_i N(u_i, \sigma_i), \tag{3}$$

where $N(u_i, \sigma_i)$ is the normal distribution with mean u_i and variance σ_i, $\Sigma_{i=1}^{k} a_i = 1$, $a_i \geq 0, i = 1, 2, \ldots, k$ and k is the order of the model.

Theorem 1 (Gaussian Mixture Model Approximation Theorem). *Finite Gaussian mixture model can approximate a non-negative Riemann integrable function in real number field by any degree of accuracy. Particularly, it is able to approximate any probability density functions by any degree of accuracy.* □

The generalized Gaussian mixture function is also defined as in (3), only with the difference that $a_i \in R$. Therefore, one can derive the following corollary. The proof of this corollary is straight according to the above theorem.

Corollary 1. *Generalized Gaussian mixture functions can approximate a continuous function in real number field by any degree of accuracy.* □

3.2 Learning Method for PNNs Based on PSO (PSO-LM)

Consider the feed-forward PNN shown in Fig. 1(b). Only the hidden neurons are simple process neurons, and the other parameters are all constants. Theoretically, the approximation capability of this PNN can be retained if the weight functions are specialized as generalized Gaussian mixture functions.

There are n input neurons. The following parameters will be optimized by PSO according to specific objectives: the order of GGMFs (k), the number of hidden neurons (m), the coefficients, means and variances of Gaussian functions corresponding to the weight functions that connect from input neurons to hidden neurons ($a_{ijl}, u_{ijl}, \sigma_{ijl}, i = 1, \ldots, n; j = 1, \ldots, m; l = 1, \ldots, k$), the thresholds of hidden neurons ($\theta_j, j = 1, \ldots, m$) and the output neuron (θ).

So, a particle in the swarm of PSO can be expressed as:

$$p = (k, m, a_{ijl}, u_{ijl}, \sigma_{ijl}, \theta_j, w_j, \theta), i = 1, \ldots, n; j = 1, \ldots, m; l = 1, \ldots, k. \quad (4)$$

Each position of the particle in the search space represents one process neural network. The PSO will search for a PNN that has appropriate structure and parameter values to optimize the objective function, which is the performance evaluation of the PNN. When the swarm of PSO converges to one position, which is supposed to be the best parameter setups of the PNN, the learning process terminates.

According to the above study, the PSO-LM for PNNs can be summarized as in Algorithm 1:

Algorithm 1. PSO-LM for PNNs

Step 1: Set up the objective function of the training as the fitness function of PSO and the generalized Gaussian mixture functions as the weight functions of PNs shown in Fig. 1(a).
Step 2: Set parameters of PSO (particle number p, w, c, search space borderlines).
Step 3: Each particle in PSO is formed according to (4).
Step 4: Run PSO.
Step 5: Stop criterion (maximum iterations or/and fixed learning precision).
Step 6: If the result is not satisfactory, go back to Step 3.

3.3 Discussions

The proposed PSO-LM has several advantages comparing with the existing learning methods for PNNs.

First, PSO-LM does not need any transformation to the input signals, which will not lose any information carried in inputs. BFE-LM extracts the principal components of the input data by using the basis function expansion, which will retain most of the information. However, the detailed information carried in inputs is the most needed information in some task. For example, in pattern recognition, two patterns might have the same first and second principal components, which will be reserved according to BFE-LM. But, it is the detailed differences carried in their third components that can distinguish them, which will be eliminated by BFE-LM. Thereafter, samples from different classes can be misclassified into the same pattern. Furthermore, the transformations also bring more computation.

Second, gradient information is no longer needed in PSO-LM. The structure and the parameters of the PNNs can be optimized simultaneously by PSO. The learning capability will be enhanced, which is guaranteed by the strong optimization capability of PSO.

Furthermore, PSO-LM is able to solve high dimensional data learning problems by using high dimensional GGMFs as the weight functions. PSO-LM can also solve multi-objectives training problems as PSO is good at doing multi-objective optimization.

The computational complexity of PSO-LM for PNNs can be expressed as $C = F \times P \times I$. F is the amount of one time feed-forward calculation of PNN which is also one time fitness calculation for PSO, P is the number of particles in PSO, and I is the maximum number of iterations. The generalization performance of PSO-LM can be verified by the following experiments.

4 Experiments

4.1 Traffic Flow Prediction

The data is from $PeMS$ system [8], which is a performance measurement system of freeway traffic from University of California. In this study, we chose traffic volume data recorded every five minutes in the time interval 10:00 to 15:00 during the weekdays from 2009/6/1 to 2009/6/21. The data is taken from five sequential loop detectors, whose serial numbers are 716810, 717907, 717898, 717896 and 717887 in District 7.

Experimental Setups. The objective is to predict the traffic flow of loop 717898, which lies in the middle of the five loops, in the next five minutes. We will use the previous five records from all the five loops to predict the next value of the third loop [3]. The data from the first two weeks which contains 120 samples is used to train the three models mentioned above, and the data of the third week which has 55 samples is used as the test data to test their generalization performances.

The parameters of the three models are listed in Table 1. The number of $db1$ wavelet basis functions chosen in BFE-LM is ten, which is decided by trial-and-error and is also the number of neurons in the first hidden layer of BFE-LM. k is the order of the GGMFs and m is the number of neurons in hidden layer of PSO-LM, which will both be decided by PSO to optimize the objective function. Finally, the activation functions of neurons are all chosen to be $purelin$ function according to experiments. Notice that $k = 2$ and $m = 13$ is decided by PSO.

Table 1. Parameter Setups For Traffic Flow Experiment

	Structures	*Activation Functions*	*Parameters*
$BPNN$	5-25-1	purelin, purelin	
$BFE-LM$	5-10-25-1	purelin, purelin	$db1$ wavelet basis
$PSO-LM$	5-m-1	purelin, purelin	k=2, m=13

The evaluation of the models depends on the following two targets:

$$MPAE = \frac{1}{n}\sum_{t=1}^{n}\frac{\mid y_t - \widehat{y}_t \mid}{y_t} \tag{5}$$

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n}(y_t - \widehat{y}_t)^2}{n}} \tag{6}$$

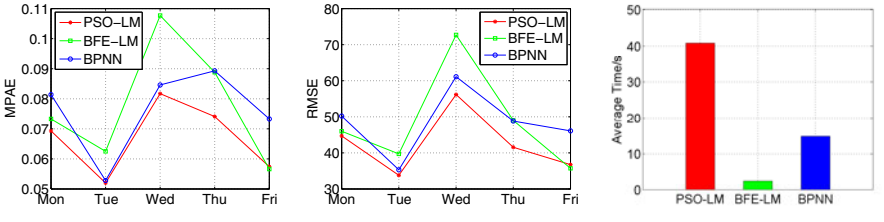
We will compare the models on each day of the weekdays independently. Each of the models ran 30 times independently, and the averaged results were recorded.

Experimental Results. The averaged results are shown in Fig. 2. As can be seen, PSO-LM outperforms BFE-LM and BPNN. According to the variance analysis, the p values of $MPAE$ and $RMSE$ are 0.013 and 0.034, which mens the improvements of the performance by PSO-LM is statistically significant.

The PSO-LM can learn the data well and predict the traffic flow more precisely, which indicate that PSO-LM can give PNNs more powerful learning capability and also good generalization capability. PSO-LM spent the most time to achieve the best performance for its global search strategy. BFE-LM consumes the least time, because the basis function expansion for the inputs retains only the principal components of the data, which makes it easier for NNs to learn. Nevertheless, BFE-LM can not guarantee good performances.

4.2 Spam Detection

In this experiment, the three models will be compared in identifying spam on a standard $Spambase$ data [9]. There are 4601 samples, each of which has 57 attributes and one



(a) MPAE (b) RMSE (c) Time Consuming

Fig. 2. Experimental Results For Traffic Flow Forecasting

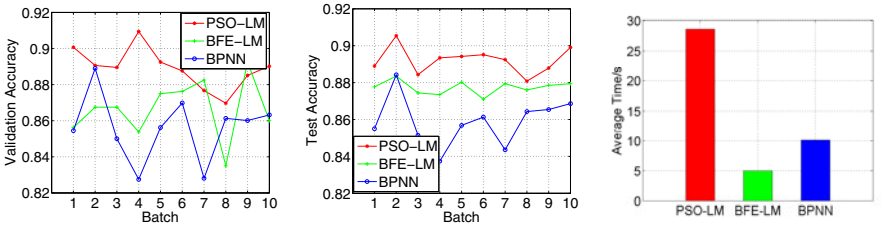
label. The label denotes whether the e-mail is considered as a spam (1) or not (0). In this study, we will use the first 48 attributes which denote the frequencies of 48 words occurring in the e-mails.

Experimental Setups. We use cross-validation in this experiment. First, 601 samples are randomly chosen to join a test data. Then, we use the rest 4000 samples to perform the standard ten times cross-validation to train and validate the models. Finally, the trained models will be tested on the test data to show their generalization capabilities. Each model will run 30 times independently to get statistical results. The parameters in this experiment are shown in Table 2. All the parameters are with the same meanings as in Table 1.

Table 2. Parameter Setups For *Spambase* Experiment

	Structures	*Activation Functions*	*Parameters*
$BPNN$	48-15-1	tansig, logsig	
$BFE-LM$	1-10-5-1	tansig, logsig	$db1$ wavelet basis
$PSO-LM$	1-m-1	purelin, logsig	k=4, m=16

Experimental Results. The averaged results are shown in Fig. 3. The validation accuracies of the ten times cross-validation are shown in Fig. 3(a). As can be seen, PSO-LM can obviously outperform the other two models, which indicates that the PSO-LM has the best learning capabilities. The test accuracies are shown in Fig. 3(b). Similarly, PSO-LM has the best generalization capability. Time consuming comparison is shown in Fig. 3(c). As the same reason as in the above experiment, the PSO-LM needs more time to gain better performance that the other models can not achieve. Furthermore, the p values of the validation and test accuracy are 0.0486 and 0.0491, which means the performance improvements made by PSO-LM are statistically significant.



(a) Validation Accuracy (b) Test Accuracy (c) Time Consuming

Fig. 3. Experimental Results For Spam Detection

5 Conclusion

This paper proposed a new learning method for process neural networks based on the Gaussian mixture weight functions and particle swarm optimization (PSO-LM). The weight functions were specified as the generalized Gaussian mixture functions. The structure and parameters in the model were then optimized by PSO. According to experiments, PSO-LM had better performance on time-series prediction and pattern recognition than BFE-LM and BPNNs. Although it needed more computations for the global search strategy of PSO, it achieved better performances that other methods can not gain.

Acknowledgement

This work is supported by National Natural Science Foundation of China (NSFC), under grant number 60875080 and 60673020, and partially financially supported by the Research Fund for the Doctoral Program of Higher Education (RFDP) in China. This work is also in part supported by the National High Technology Research and Development Program of China (863 Program), with grant number 2007AA01Z453.

References

1. He, X.G., Xu, S.H.: Process Neural Networks. Science Press, Beijing (2007)
2. He, S., Hu, C., Song, G.J., Xie, K.Q., Sun, Y.Z.: Real-Time Short-Term Traffic Flow Forecasting Based on Process Neural Network. In: Sun, F., Zhang, J., Tan, Y., Cao, J., Yu, W. (eds.) ISNN 2008, Part II. LNCS, vol. 5264, pp. 560–569. Springer, Heidelberg (2008)
3. Wu, T.S., Xie, K.Q., Song, G.J., He, X.G.: Numerical Learning Method for Process Neural Network. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5551, pp. 670–678. Springer, Heidelberg (2009)
4. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc. of the IEEE International Conference on Neural Networks 1995, pp. 1942–1948 (1995)
5. Van Den Bergh, F., Engelbrecht, A.P.: Training Product Unit Networks Using Cooperative Particle Swarm Optimization. In: Proc. of the IEEE International Joint Conference on Neural Networks 2001, vol. 1, pp. 126–131 (2001)
6. Bratton, D., Kennedy, J.: Defining a Standard for Particle Swarm Optimization. In: Proc. of the IEEE Swarm Intelligence Symposlum (SIS), pp. 120–127 (2007)
7. Yuan, L.H., Li, Z., Song, J.S.: Probabillty Density Function Approximation Using Gaussian Mixture Model. Radio Communications Technology 33(2), 20–22 (2007)
8. Freeway Performance Measurement System (PeMS), http://pems.eecs.berkeley.edu
9. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. In: School of Information and Computer Science. University of California, Irvine (2007), http://archive.ics.uci.edu/ml/datasets/Spambase

Interval Fitness Interactive Genetic Algorithms with Variational Population Size Based on Semi-supervised Learning

Xiaoyan Sun*, Jie Ren, and Dunwei Gong

School of Information and Electrical Engineering,
China University of Mining and Technology, 221116, Xuzhou, China
xysun78@126.com

Abstract. In order to alleviate user fatigue and improve the performance of interactive genetic algorithms(IGAs) in searching, we introduce a co-training semi-supervised learning(CSSL)algorithm into interval fitness IGAs with large and variational population size. The CSSL is adopted to model the user's preference so as to estimate abundant of unevaluated individuals' fitness. First, the method to select the labeled and unlabeled samples for CSSL is proposed according to the clustering results of the large size population. Combined with the approximation precision of two co-training learners, an efficient strategy for selecting high reliable unlabeled samples to label is given. Then, we adopt the CSSL mechanism to train two RBF neural networks for establishing the surrogate model with high precision and generalization. In the evolution, the surrogate model estimates individuals' fitness and it is managed to guarantee the approximation precision based on its estimation error. The proposed algorithm is applied to a fashion evolutionary design system, and the experimental results show its efficiency.

Keywords: interactive genetic algorithms, interval fitness, semi-supervised learning, surrogate model, variational population size.

1 Introduction

Interactive genetic algorithms (IGAs), proposed in mid 1980s, are effective methods to solve an optimization problem with implicit or fuzzy indices [1]. These algorithms combine traditional evolution mechanism with a user's intelligent evaluation, and the user assigns an individual's fitness rather than a function which is difficult or even impossible to be explicitly expressed. Up to now, they have been successfully applied in design of simple emotions [2], hearing aid fitting [3] and so on.

The obvious characteristic of IGAs, compared with traditional genetic algorithms, is that the user assigns an individual's fitness. Frequent interactions

* This work was supported by NSFC grant No.60775044 and Program for New Century Excellent Talents in University with granted No. NCET-07-0802.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 288–295, 2010.

result in user fatigue, therefore traditional IGAs often have a small population size and few evolutionary generations, which restrict the performance of these algorithms in exploration and their applications in complicated optimization problems. Therefore, it is necessary to develop efficiency IGAs with large population size or evolutionary generations.

In IGAs with large population size, if traditional evaluating approaches are adopted, i.e., all individuals are evaluated by a user, the user must be too fatigue and the evolution even cannot go on. Fitness estimation is usually involved in such algorithms so that the population size increases whereas user fatigue dose not increase. Generally speaking, there are two kinds of fitness estimation strategies, one is clustering-based [4] [5] [6] and the other is surrogate model-assisted[7] [8].

In the clustering-based strategy, the large size population is clustered and the center individual's fitness is evaluated by the user with a crisp value and other individuals's fitness are evaluated based on the centers. There is an assumption in such algorithms that the cluster's center is the best individual of that cluster. Obviously, it is not always true, which will result in large estimation error, misleading the direction of search and influencing such algorithms' performance. Moreover, the crisp fitness cannot reflect the user's uncertain cognition in IGAs [9].

In the surrogate model-assisted IGAs, the individual's fitness is estimated by the surrogate model instead of the user, which reduces the number of individuals evaluated by the user. Compared with the clustering-based strategy, the surrogate model-assisted algorithms have been widely developed in both crisp fitness and uncertain ones, e.g., interval fitness [10] and fuzzy random fitness [11]. However, the surrogate models are trained with supervised learning algorithms, and many learning samples are usually needed for obtaining precise model. However, these samples are obtained only through many human-computer interactions which will lead to user fatigue, violating our expectations. Therefore, we should combine new machine learning strategy into IGAs for establishing precise model with few evaluated individuals.

Semi-supervised learning(SSL) [12] is a machine learning method between supervised learning and unsupervised learning, whose learning samples include labeled data and unlabeled data. It trains initial learners with few labeled samples and use large amount of unlabeled data to improve the performance of the initial models. Because SSL requires less human effort and gives higher accuracy, it has become a topic of significant recent interest. Therefore, we introduce the SSL into IGAs to develop an IGA with interval fitness and variational population size.

Among existing semi-supervised learning methods, co-training algorithm is simple and has been successfully applied in many fields, and here it is adopted. Up to now, most researches of co-training semi-supervised(CSSL) learning devotes to classification and regression with static data [12]. Whereas, the data in IGAs dynamically changes along with evolution. Therefore, we here consider the dynamic of IGAs when combining CSSL, and mainly focus on four problems: selection strategy of labeled samples and unlabeled samples for CSSL, efficient

selection method of unlabeled samples with high reliability to label, training and application of the surrogate model, and update of the surrogate model. The first two contents can improve the computation efficiency of CSSL by combining the characteristic of IGAs. And the last two contents really improve IGA's performance through applying the results of semi-supervised learning.

This work is the extension of our previous research [6], clustering-based IGAs with variational population size, but here, the individuals' fitness is interval and CSSL is adopted to construct surrogate model.

2 Related Work

The following optimization problem is considered in this paper:

$$\begin{aligned} & \max f(x) \\ \texttt{s.t.}\ \ & x = (x_1, x_2, \ldots, x_I) \in S \\ & S = g_1 \times g_2 \times \ldots \times g_I \ . \end{aligned} \tag{1}$$

where $f(x)$ is a performance index to be optimized, and can not be expressed by an explicit function, x is an I-dimensional integer decision variable belonging to the domain S, $x_j \in g_j \ j = 1, 2, ..., I$. On condition of not causing confusion, we also denote x and S as the corresponding individual and the search space, respectively.

We adopt IGAs to solve problem (1). The uncertainty of user's cognition makes it difficult for him/her to precisely assign individuals' fitness, and we have proposed an IGA with individual's interval fitness [9].

In our previous work [6], we cluster the individuals according to the similarity of gene meaning units. The clusters' centers are denoted as $c_1(t), c_2(t), ..., c_{Nc}(t)$, where N_c is the number of clustering. In our work[7], the individual's fitness is a crisp value. In this paper, the uncertainty of user's cognition is considered and the fitness of clusters' centers are expressed with intervals, i.e., $f(c_1(t)), f(c_2(t)), ..., f(c_{N_c}(t))$ with $f(c_k(t))$ being

$$f(c_k(t)) = [\underline{f}(c_k(t)), \overline{f}(c_k(t))], k = 1, 2, ..., N_c \ . \tag{2}$$

where $\underline{f}(c_k(t))$ and $\overline{f}(c_k(t))$ are the lower limit and the upper limit of the user's evaluation on the k-th center $c_k(t)$, respectively.

Combined the surrogate model-assisted algorithm with the idea of variational population size, we adopts CSSL to establish surrogate model with high precision based on the evaluated clusters' centers and other unevaluated individuals.

3 Obtainment of Surrogate Model Based on CSSL

The surrogate model which learns the user's cognition in IGAs is established by using CSSL. Labeled samples and unlabeled samples are selected combined with the characteristic of IGAs. Then according to the manifold assumption of CSSL and clustering results presented in literiture[6], we present a new strategy

for selecting many reliable unlabeled samples in once iteration when applying CSSL. The training samples and surrogate model are both updated along with the evolution to guarantee the approximation performance.

3.1 Construction of Surrogate Model with CSSL

Two issues are considered here for constructing surrogate models of IGAs with CSSL. One is the selection of training samples for CSSL, e.g., labeled samples and unlabeled ones. And the other is the building of the surrogate model.

In this paper, we establish surrogate model based on the initial population by use of CSSL with the selected learning samples. We denote the initial population as $P(1) = \{x_1, x_2, ..., x_{N_1}\}$, and cluster it using the approach presented in [6], then obtain the clusters' centers as $C(1) = \{c_1(1), c_2(1), ..., c_{N_c}(1)\}$. The user evaluates these clusters' centers and gives their fitness $f(c_k(t)) = [\underline{f}(c_k(t)), \overline{f}(c_k(t))], k = 1, 2, ..., N_c$. The labeled data set is expressed as $L(1) = \{(c_k(1), f(c_k(1)))|k = 1, 2, ..., N_c(1)\}$, the unlabeled one is $U(1) = P(1)\backslash C(1)$. In the subsequent evolutions, these two sets are dynamically updated due to continuously generated individuals and varied user preference.

With $L(1)$, we use CSSL to generate surrogate model to approximate the user's cognition. Because the individuals' fitness evaluated by the user are intervals, to construct surrogate model is just to approximate intervals. In order to guarantee the efficiency of CSSL[12], two regressors with different structure or parameters, h_1 and h_2, should be selected. In this paper, we adopt two RBF neural networks with different structures as regressors and each approximates an interval. Therefore, the outputs of each regressor are two, i.e., the upper limit and lower limit of an interval.

The $L(1)$ is firstly used to train h_1 and h_2 . Because the training samples of h_1 and h_2 change in the training process, we denote the training samples set of h_1 and h_2 as $L_1(t)$ and $L_2(t)$ with size $|L_1(t)|$ and $|L_2(t)|$, where $L_i(t) = \{(c_k^i(t), f(c_k^i(t)))|k = 1, 2, ..., |L_i(t)|\}$. For $c_k^i(t)$, the two outputs of a regressor are $\underline{\widehat{f}_i}(c_k^i(t))$ and $\overline{\widehat{f}_i}(c_k^i(t)), i = 1, 2$. To assess the performance of these regressors, the MSE of each regressor on an interval is defined as follows.

$$e_i = \frac{1}{2|L_i(t)|}\sqrt{\sum_{k=1}^{|L_i(t)|}(\underline{e}_i^2(c_k^i(t)) + \overline{e}_i^2(c_k^i(t)))}, i = 1, 2 \ . \tag{3}$$

where $\underline{e}_i(c_k(t)) = \underline{\widehat{f}}_i(c_k(t)) - \underline{f}(c_k(t)), \overline{e}_i(c_k(t)) = \overline{\widehat{f}}_i(c_k(t)) - \overline{f}(c_k(t))$. Small e_i means that the $i-th$ regressor has well approximation performance.

3.2 Identification and Selection of Reliable Unlabeled Samples

It is critical for identifying and selecting reliable unlabeled samples to label for refining regressors in CSSL. Based on the influence on the precision of the surrogate model after adding unlabeled samples, Zhou presented a method of selecting reliable samples [12]. But this method only selects one unlabeled sample

to label, which results in high computational complexity. We here still adopt that idea and propose a new strategy of selecting many reliable samples once based on manifold assumptions and clustering results, improving the efficiency of CSSL.

The j-th cluster's individuals not being evaluated by the user are denoted as $U_j = \{x_k^j\}$, and the unlabeled set is $U = \bigcup_{j=1}^{N_c(t)} U_j$. For the individual x_k^j in U_j, its estimated fitness by the h_i is $\widehat{f}_i(x_k^j(t)) = [\underline{\widehat{f}_i}(x_k^j(t)), \overline{\widehat{f}_i}(x_k^j(t))]$. The $\{x_k^j, \widehat{f}_i(x_k^j)|k = 1, 2, ...|U_j|\}$ are added to $L_i(t)$ for retraining h_i. The retrained regressor is $h_i^{'}$. Then the MSE's variations of $h_i^{'}$ and h_i is calculated as $\Delta_i(U_j) = e_i - e_i^{'}, i = 1, 2$. Where e_i and $e_i^{'}$ are calculated from Eq.(3). If $\Delta_1(U_j) > 0$, it means that the precision of h_1 increases after adding unlabeled samples U_j into the training set of h_1. Similarly, if $\Delta_2(U_j) > 0$, it means that the precision of h_2 increases after adding unlabeled samples U_j into the training set of h_2. Those individuals and their estimations in one cluster which make the largest $\Delta_1(U_j)$ $(\Delta_2(U_j))$ are reliable samples. These samples are added to $L_2(t)((L_1(t)))$ and deleted from U. The updated $L_1(t)$ and $L_2(t)$ are used for retraining h_1 and h_2. The above process is repeated until $U = \phi$.

When the above process has finished, h_1 and h_2 estimated all unevaluated individuals. For individual $x(k)$, the average outputs of h_1 and h_2 on it is calculated as the estimated fitness,i.e.,

$$\widehat{f}(x_k) = \frac{\widehat{f}_1(x_k) + \widehat{f}_2(x_k)}{2} = [\frac{\underline{\widehat{f}_1}(x_k) + \underline{\widehat{f}_2}(x_k)}{2}, \frac{\overline{\widehat{f}_1}(x_k) + \overline{\widehat{f}_2}(x_k)}{2}] \ . \tag{4}$$

3.3 Application and Updates of Surrogate Model

The surrogate model, expressed in Eq.(5), is obtained based on a few evaluated individuals generated in the initial population, and it is used to estimate all unevaluated individuals' fitness. Along with the evolution, new individuals are generated with genetic operators and the user preferred scope will change, therefore, the surrogate model must be updated when it is used in the subsequent evolutions. To perform this, there are two problems must be considered, one is when to update the surrogate model and the other is how to obtain the new training samples, i.e., how to update the learning samples of CSSL.

The population is clustered and all centers are displayed to the user together with their estimated fitness with Eq.(5). If the individual's estimation deviates from the user's preference, the user reevaluates it. It is assumed that the user reevaluate (Q) individuals with (Q) being constant, and these individuals are denoted as $c_r(t), r = 1, 2, ..., Q$, their reevaluated fitness is $f(c_r(t)) = [\underline{f}(c_r(t)), \overline{f}(c_r(t))]$. When the difference between the user's reevaluation and the estimation is big, the surrogate model needs to be updated at this time. Therefore, the update condition is as follows.

$$\frac{1}{Q}\sqrt{(\underline{f}(c_r(t)) - \underline{\widehat{f}}(c_r(t)))^2 + (\overline{f}(c_r(t)) - \overline{\widehat{f}}(c_r(t)))^2} \geq f^0 \ . \tag{5}$$

Where f^0 is a threshold.

In every generation, the user reevaluates some individuals' fitness which represent the current cognition of the user, and therefore we reserve these data to be new labeled samples for updating the surrogate model. Supposing that the surrogate model is updated in T generation, then the labeled sample set is $L(t) = \bigcup_{t=2}^{T}\{c_r(t), f(c_r(t))|r = 1, 2, ..., Q\}$.

4 Application in Fashion Evolutionary Design System

4.1 Back Ground

Fashion evolutionary design system is a typical application platform for interactive genetic algorithms. Therefore, the proposed algorithms (shorted as VPS-CSSL)is applied to a fashion evolutionary design system to show its efficiency by comparing it with the neural network based surrogate model of interval fitness IGAs (shorted as IGA-IIFSM)[10] and IGAs with variational population size (shorted as VPS)[6].

The genetic operators and control parameters of the three algorithms are set to be same. The upper limit and the lower limit of an individual's interval fitness are within $[0, 1000]$. The initial population size of VPS-CSSL, IGA-IIFSM and VPS are 200, 10 and 200.Tournament selection with size two, one-point crossover and one-point mutation operators are adopted with their probabilities being 0.6 and 0.02, respectively. $Q = 2$, $f_0 = 150$. Each experiment runs for 5 times and the average of all results is compared. Every generation 10 suits are displayed to the user on the interface. An individual's phenotype is a suit composed of the styles of coat and skirt and their color. There are 32 styles and 16 colors of coats and skirts respectively. Then, the decision variable space is $2^5 * 2^5 * 2^4 * 2^4 = 262144$.

Fig. 1. (a)Interface of human-computer interaction.(b)The optimal result

The corresponding interactive interface of VPS-CSSL is shown in Fig. 1(a). The evolution repeated by performing this interface until the user finds his/her satisfactory solutions, e.g., shown in Fig.1(b).

4.2 Results and Analysis

This algorithm is an improvement of VPS, and we do some comparisons of VPS and VPS-CSSL to illustrate the performance of our algorithm. On condition of finding the optimal suit shown in fig.1(b), we compare these two algorithms from

Table 1. Comparisons of VPS-CSSL and VPS

		♯ of individuals evaluated by user	time-consuming	generations
average	VPS-CSSL	42.4	6'38"	17.2
	VPS	132	13'14"	13.2

3 aspects, i.e., number of individuals evaluated by the user, time-consuming and generations, and the statistical results are listed in Table 1.

It can be observed from Table 1 that the average number of individuals being evaluated by the user of our algorithm is 42.4, much smaller than that of VPS, i.e., 132. The average time-consuming of the user's evaluations of our algorithm is 6'38", much less than that of VPS,13'14". These two results show that our algorithm greatly alleviates user fatigue. Whereas, it is worth noting that the evolutionary generations of VPS-CSSL are 17.2, larger than VPS's 13.2. The reason may be that the surrogate model has little error and can not learn the user's cognition absolutely. However, the user fatigue does not increase. To sum up, compared with VPS, our algorithm can find the optimal solution with less user fatigue.

Then, we compare VPS-CSSL with IGA-IIFSM in four aspects and the results are listed in Table 2.

Table 2. Comparisons of VPS-CSSL and IGA-IIFSM

		♯ of user evaluated individuals	generations	time-consuming	♯ of searched individuals
average	VPS-CSSL	42.4	17.2	6'38"	2583.6
	IGA-IIFSM	150.4	51.4	10'21"	312.8

Similarly with the analysis of Table 1, we can draw the conclusion that our algorithm has more opportunities to find the optimal solution because variational population size is adopted in our algorithm, and it improves the IGAs' performance in searching.

To sum up, our algorithm greatly alleviates user fatigue and has more opportunities to find the optimal solutions due to the combination of CSSL and IGAs with variational population size.

5 Conclusions

Aiming at solving interval fitness IGAs with variational population size, we use a little information to establish surrogate model through semi-supervised learning.

And the surrogate model estimates other individuals' fitness in clusters and in subsequent evolutions. The user reevaluates some individuals' fitness, forming updated data samples. In order to guarantee the precision of the surrogate model, we update it in an appropriate time. The proposed algorithm is successfully applied in a fashion evolutionary design system, and experimental results show its advantageous in searching and alleviating user fatigue. How to solve IGAs with large population size and an individual's fuzzy fitness is an issue to be further studied.

References

1. Dawkins, R.: The Blind Watchmaker. Longman, UK (1986)
2. Dozier, G., Carnahan, B., Seals, C., et al.: An Interactive Distributed Evolutionary Algorithm (IDEA) for Design. In: The IEEE International Conference on Systems, Man and Cybernetics, pp. 418–422. IEEE Press, New York (2005)
3. Takagi, H., Ohsaki, M.: Interactive Evolutionary Computation-based Hearing Aid Fitting. J. IEEE Transactions on Evolutionary Computation 11, 414–427 (2007)
4. Kim, H.S., Cho, S.B.: An Efficient Genetic Algorithm with Less Fitness Evaluation by Clustering. In: IEEE Congress on Evolutionary Computation, pp. 887–894. IEEE Press, New York (2001)
5. Gong, D.W., Yuan, J., Ma, X.P.: Interactive Genetic Algorithms with Large Population Size. In: Proceedings of the 2008 IEEE Congress on Evolutionary Computation, CEC 2008, pp. 1678–1685. IEEE Press, New York (2008)
6. Ren, J., Gong, D.W.: Interactive Genetic Algorithms with Variational Population Size. In: Huang, D.-S., Jo, K.-H., Lee, H.-H., Kang, H.-J., Bevilacqua, V. (eds.) ICIC 2009. LNCS, vol. 5755, pp. 64–73. Springer, Heidelberg (2009)
7. Llora, X., Sastry, K., Goldberg, D.E., et al.: Combating User Fatigue in Igas: Partial Ordering, Support Vector Machines, and Synthetic Fitness, Illinois Genetic Algorithms Lab. University of Illinois, Urbana-Champaign (2005)
8. Ecemis, I., Bonabeau, E., Ashburn, T.: Interactive Estimation of Agent-Based Financial Markets Models: Modularity and Learning. In: Genetic and Evolutionary Computation Conference, pp. 1897–1904. ACM Press, New York (2005)
9. Gong, D.W., Guo, G.S.: Interactive Genetic Algorithms with Interval Fitness of Evolutionary Individuals. In: Dynamics of Continuous, Discrete and Impulsive Systems. Series B, pp. 446–450 (2007)
10. Gong, D., Sun, X.: Surrogate Models Based on Individual's Interval Fitness in Interactive Genetic Algorithms. Chinese Journal of Electronics, 689–694 (2009)
11. Sun, X.Y., Gong, D.W., Ma, X.P.: Directed Fuzzy Graph Based Surrogate Model Assisted Interactive Genetic Algorithms with Uncertain Individual's Fitness. In: 2009 IEEE Congress on Evolutionary Computation, CEC 2009, pp. 2395–2402. IEEE Press, New York (2009)
12. Zhou, Z.H., Li, M.: Semi-Supervised Regression with Co-Training Style Algorithms. IEEE Transactions on Knowledge and Data Engineering 19, 1479–1493 (2007)

Research on One-Dimensional Chaos Maps for Fuzzy Optimal Selection Neural Network

Tao Ding[1,*], Hongfei Xiao[2], and Jinbao Liu[3]

[1] College of Quality & Safety Engineering, China Jiliang University, Hangzhou, Zhejiang Province 310018, China
dingtao@cjlu.edu.cn
[2] School of Automation, Hangzhou Dianzi University, Hangzhou, Zhejiang Province 310018, China
xiaohf@hdu.edu.cn
[3] Zhejiang Tongji Vocational College of Science and Technology, Hangzhou, Zhejiang Province 311231, China
liujinbao75@163.com

Abstract. For solving the optimization problems with slow convergence speed and local optimum in fuzzy optimal selection neural network, this paper applies the chaos optimization algorithm by using a chaos variable from one-dimensional iterative map to optimize the network weight. For selecting the reasonable chaos variable, multi one-dimensional chaos maps, such as Logistic Map, Sine Map, Cosine Map and Cubic Map, are researched and compared. To verify feasibility of one-dimensional chaos map for fuzzy optimal selection neural network in the practical application, the case of Yamadu Hydrological Station located in Yili River for annual runoff forecast is analyzed and discussed. The results show that the chaos optimization algorithm is an efficient learning algorithm which has the advantage of speed convergence and high precision for fuzzy optimal selection neural network.

Keywords: chaos map; fuzzy optimal selection; neural network; optimization.

1 Introduction

The theory of fuzzy optimal selection has been widely applied in many engineering field since it was proposed by professor Chen [1] in 1990. The papers [2,3] further combined fuzzy optimal selection with neural network, and the reasonable method of determining network topological structure and adjusting amount equation of weight were presented in 1997. Recently, there has been a growing interest in combining fuzzy optimal selection, BP neural network and genetic algorithm [4,5,6]. In general, the weights of fuzzy optimal selection neural network are calculated by gradient-descended algorithm in the above research. Therefore, sometimes there exist the problems of slow convergence speed and local optimum in solving optimization problems of some multi-modal functions. Since chaos movement has the characteristics of the

* This work is partially supported by ZJNSF Grant # Y505066 to Tao Ding.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 296–302, 2010.

interior stochastic, the ergodicity and the regulation, especially the ergodicity which can be used as the optimal mechanism to avoid local optimum, the paper applies the chaotic optimization algorithm by using a chaos variable from one-dimensional iterative map to optimize the network parameters. To verify feasibility of the chaos optimization algorithm, the case of Yamadu Hydrological Station located in Yili River for annual runoff forecast is analyzed and discussed.

2 Fuzzy Optimal Selection Neural Network

Cause-effect relationship is one of expressive form of prevalent contact and mutual restriction in objective world and in nature the change of some phenomena or factors will arouse other phenomena or factors produce. Therefore, we assume that y (forecast object) represents some phenomena affected by multi-factors, and x_i (forecast factors) represent a variety of factors that effect y. Apparently forecast object y is the result influenced by multi-factors.

Assuming that there are n forecast object that constitute sample set, and their feature vector are $y = (y_1, y_2, \ldots, y_n)$. By using (1), we get relative membership degree matrix of forecast object to fuzzy set $\underline{A}$ in [7].

$$\begin{aligned} \left({}_y r_j\right) &= \left({}_y r_1, {}_y r_2, \cdots, {}_y r_n\right) \\ {}_y r_j &= (y_j - \min y_j)/(\max y_j - \min y_j) \end{aligned} \tag{1}$$

Each forecast object parallels m feature values, and for n samples, feature values matrix of forecast factors may be expressed as

$$X = \left(x_{ij}\right), i = 1,2,\cdots,m; j = 1,2,\cdots,n \tag{2}$$

where x_{ij} is feature value of sample jth to factor ith. Due to physical dimension of m feature values of forecast factors is different, matrix (2) should be normalized. If forecast factors has positive correlation with forecast object y, normalized equation (3) are adopted, otherwise, (4) are adopted.

$${}_x r_{ij} = (x_{ij} - x_{i\min})/(x_{i\max} - x_{i\min}) \tag{3}$$

$${}_x r_{ij} = (x_{i\max} - x_{ij})/(x_{i\max} - x_{i\min}) \tag{4}$$

$$x_{i\max} = \bigvee_{j=1}^{n} x_{ij}, x_{i\min} = \bigwedge_{j=1}^{n} x_{ij}$$

After the above procedure, matrix (2) is transformed into relative membership degree matrix of feature values in section [0 1]:

$${}_x R = ({}_x r_{ij}), i = 1,2,\cdots,m; j = 1,2,\cdots,n \tag{5}$$

According to Chen (2002), the fuzzy optimal selection network can be constructed as 3 layers with m input neurons, k hidden neurons and a scalar output. The sketch of network is depicted in Fig. 1.

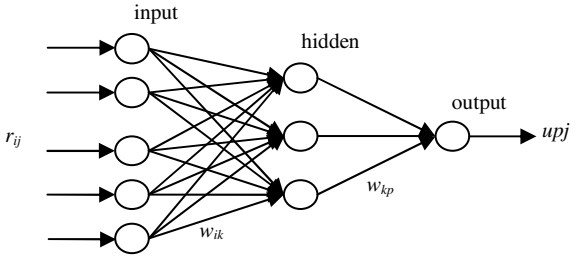


Fig. 1. Sketch of fuzzy optimal selection neural network

For the neuron i of input layer, information of node i directly is transferred to hidden layer, so its input and output is equal.

$$u_{ij} = r_{ij} \tag{6}$$

For the neuron k of hidden layer, its input can be expressed as (7)

$$I_{kj} = \sum_{i=1}^{m} w_{ik} r_{ij} \tag{7}$$

where w_{ik} is the weight between input node i and hidden node k. Stimulation function of hidden node adopts sigmoid function, its output is

$$u_{kj} = 1 + \left[\left(\sum_{i=1}^{m} w_{ik} r_{ij} \right)^{-1} - 1 \right]^{-2} \tag{8}$$

For output layer, there is only one neuron, its input is

$$I_{pj} = \sum_{k=1}^{l} w_{kp} u_{kj} \tag{9}$$

where w_{kp} is the weight between hidden node k and output node p. And Stimulation function of node p also adopts sigmoid function, its output is

$$u_{pj} = 1 + \left[\left(\sum_{k=1}^{l} w_{kp} u_{kj} \right)^{-1} - 1 \right]^{-2} \tag{10}$$

The network output u_{pj} is the response of fuzzy optimal selection network to input sample $\left({}_{x} r_{1j}, {}_{x} r_{2j}, \cdots, {}_{x} r_{mj} \right)$. Assuming that expected output of sample j is $M\left(u_{pj}\right)$, thus its mean square errors of n samples is

$$E = \frac{1}{2n} \sum_{j=1}^{n} E_j = \frac{1}{2n} \sum_{j=1}^{n} \left[u_{pj} - M\left(u_{pj}\right) \right]^2 \tag{11}$$

3 One-Dimensional Chaos Maps for Neural Network

The studies [8, 9] reveal chaos optimization algorithm has high efficiency and high precision in the seeking optimization process, especially in solving optimization

problems of large space and multi-variable references, and its theory is simple and program easily. Therefore, this paper applies the chaos algorithm to optimize network parameters w_{ik} and w_{kp}. For simplifying expression, w_{ik} and w_{kp} are expressed as w_i. In order to select the reasonable chaos variable, multi one-dimensional chaos maps, such as Logistic Map, Sine Map, Cosine Map and Cubic Map, are discussed and compared. The above chaos maps are expressed as

$$x_{k+1} = \mu x_k (1 - x_k), \mu = 4$$
$$x_{k+1} = \mu \sin(\pi x_k), \mu = 0.99$$
$$x_{k+1} = \mu \cos(\pi |x_k - 0.5|), \mu = 0.99$$
$$x_{k+1} = \mu x_k (1 - x_k^2), \mu = 2.59$$

Consider the following optimization about the minimum of the square errors functions in (11).

$$min\ E(w_i) = min \left\{ \frac{1}{2n} \sum_{j=1}^{n} \left[u_{pj} - M\left(u_{pj}\right) \right]^2 \right\}\ w_i \in [a_i, b_i], i = 1,2,\cdots,n \tag{12}$$

where variable w_i belongs to $[a_i, b_i]$, i is variable number. The chaos variables come from the above four one-dimensional iterative logistic maps.

Suppose k is iterative numbers, and the basic steps of the chaos optimization algorithm based chaos variable are expressed as follows:

Step 1 Initialization of the algorithm. Let k =0, $x_i^k = x_i(0)$, $x_i^* = x_i(0)$, E^* is a big positive number, and there are i chaos variables x_i (i = 1, 2, …, n) from chaos map when random $x_i(0)$ is given as the initial value of the self-map.

Step 2 Transformation of chaos variables in optimization design region of the variable. The region [0 1] of ith chaos variable can be transformed into the optimization design region of the variable $[a_i, b_i]$ by using (13).

$$w_i^k = a_i^r + x_i^k (b_i - a_i) \tag{13}$$

Step 3 Iterating searching of chaos variables. If $E(w_i^k) < E^*$, then $E^* = E(w_i^k)$, $w_i^* = w_i^k$, $x_i^* = x_i^k$; If $E(w_i^k) \geq E^*$, then continue next iterating search.

Step 4 $k = k + 1$, $x_i^k = f(x_i^k)$.

Step 5 Repetition of step 2 ~ step 4 until E^* keeps unchanged within enough iterating times.

Here, this paper applies chaos variable from Logistic Map, Sine Map, Cosine Map and Cubic Map to optimize the neural network constructed in the following case study. Each chaos map for optimizing network runs 10 times, and initial values of each optimization are given randomly. The compared results indexed by iterative times k and mean square error (MSE) show that the above maps all attain the expected precision and the efficiency is little difference among them. But the initial values of chaos maps have obvious effect on convergence speed.

Table 1. Compared results of one-dimensional chaos maps by iterative times k and MSE

Network learning	Logistic Map		Sine Map		Cosine Map		Cubic Map	
	k	MSE	k	MSE	k	MSE	k	MSE
1	174	0.0113	3167	0.0118	3544	0.0092	842	0.0114
2	4149	0.0115	2105	0.0102	1348	0.0094	4641	0.0103
3	3606	0.0107	2015	0.0093	18	0.0117	11728	0.0108
4	3553	0.0118	4264	0.0104	3671	0.0100	3121	0.0097
5	2652	0.0113	2668	0.0114	3480	0.0110	4625	0.0119
6	590	0.0071	2476	0.0072	2356	0.0096	3762	0.0119
7	2447	0.0117	6800	0.0119	1907	0.0095	291	0.0101
8	509	0.0079	502	0.0117	1124	0.0111	252	0.0116
9	327	0.0112	2402	0.0095	4347	0.0112	9207	0.0088
10	252	0.0071	2469	0.0095	576	0.0101	1146	0.0110
Average	1826	0.0102	2887	0.0103	2237	0.0103	3962	0.0108

4 Case Study

To verify feasibility of the chaos optimization algorithm for fuzzy optimal selection neural network in the practical application, the case of Yamadu Hydrological Station for annual runoff forecast is analyzed and discussed. The observed annual runoff quantities of 23 years and its relevant 4 forecast factors are listed in table 2 where factor x_1 is aggregate rainfall volume from last Nov. to this Mar of Yili weather

Table 2. Observed annual runoff and relevant factors of Yamadu Hydrological Station

Sample	x_1	x_2	x_3	x_4	y
1	114.6	1.10	0.71	85	346
2	132.4	0.97	0.54	73	410
3	103.5	0.96	0.66	67	385
4	179.3	0.88	0.57	87	446
5	92.7	1.15	0.44	154	300
6	115.0	0.74	0.65	252	453
7	163.6	0.85	0.58	220	495
8	139.5	0.70	0.59	217	478
9	76.7	0.95	0.51	162	341
10	42.1	1.08	0.47	110	326
11	77.8	1.19	0.57	91	364
12	100.6	0.82	0.59	83	456
13	55.3	0.96	0.40	69	300
14	89.8	0.96	0.39	105	314
15	78.5	0.89	0.44	94	280
16	29.8	0.83	0.49	120	289
17	248.6	0.79	0.50	147	483
18	64.9	0.59	0.50	167	402
19	95.7	1.02	0.48	160	384
20	152.1	1.04	0.49	77	433
(21)	121.8	0.83	0.60	140	401
(22)	81.0	1.08	0.54	96	336
(23)	90.0	0.95	0.43	89	301

station, factor x_2 is lunar average zonal circulation index in last Aug. at Europe and Asia region, factor x_3 is meridianal index of Europe and Asia region in last May, factor x_4 is 2800MHz sun radio jet stream of last Jun. In this case, the former 20 years data are used for determining model parameters, and latter 3 year data are used for verifying model.

According to samples data 1-20 in table 2, the correlation coefficients of annual runoff y to forecast factor x_i can be calculated, viz. ρ_1=0.80, ρ_2=-0.63, ρ_3=0.44 and ρ_4=0.40. So factors x_1, x_3, x_4 use (3) and x_2 uses (4) for normalizing. We take the former 20 years data as training samples and network topologic structure can be constructed as 3-layer network: 4 nodes input, one node output and 4 nodes hidden layer on experience. The one-dimensional chaos maps are applied to training the fuzzy optimal selection network, and weights (w_{ik}), (w_{kp}) are obtained.

At last, we use optimized weights to forecast last 3 year annual runoff in Yamadu. The forecast results are listed in table 3. From relative error of forecast, we can see that forecast precision is satisfying, and it proves that the one-dimensional chaos maps for fuzzy optimal selection neural network are valid.

Table 3. Runoff and its forecast of the samples for testing

Sample	Observed (m^3/s)	Forecast (m^3/s)	Relative error (%)
21	401	440	9.73
22	336	315	6.25
23	301	312	3.65

5 Discussion

This paper applies chaos variable from Logistic Map, Sine Map, Cosine Map and Cubic Map to optimize the weights of fuzzy optimal selection network. From the compared results of the above chaos maps show that they all attain the expected precision and the efficiency is little difference among them. But the initial values of chaos maps have obvious effect on convergence speed. The chaos optimization algorithm has many advantages, such as fast search, precise result, and convenience. To verify feasibility of the chaos optimization algorithm for fuzzy optimal selection neural network in the practical application, the case of Yamadu Hydrological Station located in Yili River for annual runoff forecast is analyzed and discussed. The results show that the chaos optimization algorithm is an efficient learning algorithm for fuzzy optimal selection neural network.

References

1. Chen, S., Zhao, Y.: Fuzzy Optimum Selection Theory and Model. Fuzzy System and Mathematics 2, 10–14 (1990)
2. Chen, S.: Multiobjective Decision-making Theory and Application of Neural Network with Fuzzy Optimum Selection. Journal of Dalian University of Technology 37(6), 693–698 (1997)

3. Chen, S., Nie, X., Zhu, W.: A Model of Fuzzy Optimization Neural Networks and Its Application. Advances in Water Science 10(1), 69–74 (1999)
4. Guo, Y., Wang, D.: Intelligent Forecasting Mode and Approach of Mid and Long Term Intelligent Hydrological Forecasting. Engineering Science 8(7), 30–35 (2006)
5. Liu, Y., Yuan, J., Zhou, H.: Research on Application of Fuzzy Optimization Neural Network Model to Medium-term and Long-term Runoff Forecast. Journal of Dalian University of Technology 48(3), 411–416 (2008)
6. Ji, H., Chao, L., Chen, S.: Fuzzy Optimization BP Neural Network Model Based on Genetic Algorithm for Ice Forecasting. China Rural Water and Hydropower (1), 5–10 (2009)
7. Chen, S.: Fuzzy Recognition Theory and Application for Complex Water Resources System Optimization. Jilin University Press, Changchun (2002)
8. Li, B., Jiang, W.: Chaos Optimization Method and Its Application. Control Theroy and Applications 14(4), 613–615 (1997)
9. Zhang, T., Wang, H., Wang, Z.: Mutative Scale Chaos Optimization Algorithm and Its Application. Control and Decision 14(3), 285–288 (1999)

Edited Nearest Neighbor Rule for Improving Neural Networks Classifications

R. Alejo[1], J.M. Sotoca[1], R.M. Valdovinos[2], and P. Toribio[3]

[1] Institute of New Imaging Technologies
Dept. Llenguatges i Sistemes Informátics, Universitat Jaume I
Av. Sos Baynat s/n, 12071 Castelló de la Plana (Spain)
[2] Centro Universitario UAEM Valle de Chalco, Universidad Autónoma del Estado de México
Hermenegildo Galena No.3, Col. Ma. Isabel, 56615 Valle de Chalco (Mexico)
[3] Centro Universitario UAEM Atlacomulco, Universidad Autónoma del Estado de México
Km. 60 Carretera Toluca - Atlacomulco, 50450 Atlacomulco (Mexico)

Abstract. The quality and size of the training data sets is a critical stage on the ability of the artificial neural networks to generalize the characteristics of the training examples. Several approaches are focused to form training data sets by identification of border examples or core examples with the aim to improve the accuracy of network classification and generalization. However, a refinement of data sets by the elimination of outliers examples may increase the accuracy too. In this paper, we analyze the use of different editing schemes based on nearest neighbor rule on the most popular neural networks architectures.

Keywords: neural networks; editing techniques; reduction training set; accuracy.

1 Introduction

Artificial neural networks (ANN) are computational models that have become a popular tool used in remote sensing data analysis, the computer-aided medical diagnosis and the identification of microbiological taxa. However, it is well know that in supervised classification the maximum accuracy depends on the quality of the training data set. In this sense, several approaches in the area of neural networks are towards to training data selection with the aim to improve the performance of the network in terms of speed, computational requirements and classification accuracy.

Most research on this topic has traditionally focused to obtain reduced training data sets (TS) by the identification of two kinds of samples: *i)* core training patterns and *ii)* border training patterns [1,2]. However, the outliers or noise may introduce a false decision boundary. Consequently, the training data extracted may also be subject to refinement intelligent procedures.

An outlier has traditionally been defined as a prototype that does not follow the same model as the rest of the data [3]. In this context, some approaches to allow remove outliers from the original training set and also cleaning possibles overlapping among classes. This strategy has generality been referred as to editing [4].

The general idea behind almost any editing procedure consists of estimating the true classification of prototypes in the TS to retain only those which are correctly labeled.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 303–310, 2010.

The first proposal to select a representative subset of prototypes for a further nearest neighbour classification corresponds to Wilson editing algorithm [5], in which a k-NN classifier is used to retain in the TS only good samples (that is, training samples that are correctly classified by the k-NN rule).

In the present work, we study the use of several editing schemes proposed in the literature and their effects on the classification performance of three supervised artificial neural networks. This study mainly tries to show how these methods used in the nearest neighbour rule improve the classification accuracy and generalization of the most popular neural networks architectures. In order to accomplish this, we experiment with two class data sets and multiclass data sets.

The structure of the paper is as follows. Section 2 presents the learning algorithms. In Section 3, we briefly describe the editing algorithms used to reduce the training data sets. Section 4 consists of experiments on real data sets and an exhaustive discussion of results. Finally, we will conclude the main remarks and outline some directions for future work in Section 5.

2 The Classifiers

In this section, we briefly describe the classifiers selected for the present experimental study. All these algorithms work under the assumption that there exists a set of n previously labeled examples (training set, TS), say $\mathbf{X} = \{(\mathbf{x}_1, \omega_1), (\mathbf{x}_2, \omega_2), \ldots, (\mathbf{x}_n, \omega_n)\}$, where each element has an attribute vector $\mathbf{x}_i$ and a class label ω_i.

2.1 Multilayer Perceptron

The multilayer perceptron (MLP) neural network [6] usually comprises one input layer, one or more hidden layers, and one output layer. Input nodes correspond to features, hidden layers are used for computations, and output layers are the problem classes. A neuron is the elemental unit of each layer. It computes the weighted sum of its inputs, adds a bias term and drives the result thought a generally nonlinear (commonly, sigmoid) activation function to produce a single output.

The most popular training algorithm for MLP is the backpropagation, which takes a set of training instances for the learning process. For the given feedforward network, the weights are initialized to small random numbers. Each training instance is passed through the network and the output from each unit is computed. The target output is compared with the output estimated by the network to calculate the error, which is fed back through the network. To adjust the weights, backpropagation uses gradient descent to minimize the squared error between the target output and the computed output. At each unit in the network, starting from the output unit and moving down to the hidden units, its error value is used to adjust weights of its connections so as to reduce the error. This process of adjusting the weights is repeated for a fixed number of times or until the error is small or it cannot be reduced.

2.2 Radial Basis Function

The radial basis function (RBF) [7] neural network, which has three layers, can be seen as an especial kind of multilayer feedforward networks. Each unit in the hidden layer

employs a radial basis function, such as Gaussian kernel, as the activation function. The output units implement a weighted sum of hidden unit outputs. The input into an RBF network is nonlinear. The output is linear. The kernel is centered at the point specified by the weight vector associated with the unit. Both the positions and the widths of these kernels are learned from training instances. Each output unit implements a linear combination of these radial basis functions.

An RBF is trained to learn the centers and widths of the Gaussian function for hidden units, and then to adjust weights in the regression model that is used at the output unit. To learn the centers of the Gaussian functions, the k-means algorithm can be used, obtaining k Gaussian functions for each attribute in the instance. After the parameters for the Gaussian function at the hidden units have been found, the weights from these units to the output unit are adjusted using a linear regression model.

2.3 Support Vector Machine

Support vector machines (SVMs) [8] are a set of related supervised learning methods used for classification and regression. They belong to a family of generalized linear classifiers. A special property of SVMs is that they simultaneously minimize the empirical classification error and maximize the geometric margin; hence they are also known as maximum margin classifiers.

SVMs map input vectors to a higher dimensional space where a maximal separating hyperplane is constructed. Two parallel hyperplanes are constructed on each side of the hyperplane that separates the data. The separating hyperplane is the hyperplane that maximizes the distance between the two parallel hyperplanes. An assumption is made that the larger the margin or distance between these parallel hyperplanes the better the generalization error of the classifier will be.

3 Algorithms to Select Training Samples

The present section describes the procedures for handling outliers in a TS. This alternatives are based on the employment of a surrounding neighborhood to obtain a filtered TS, that is, to detect and remove outliers from the TS.

3.1 Edited Nearest Neighbor Rule

Wilson [5] developed the Edited Nearest Neighbor (ENN) algorithm in which **S** starts out the same as TS, and then each instance in S is removed if it does not agree with the majority of its k nearest neighbors (with k=3, typically). This edits out noisy instances as well as close border cases, leaving smoother decision boundaries. It also retains all internal points, which keeps it from reducing the storage requirements as much as most other reduction algorithms. Algorithmically, the ENN scheme can be expressed as follows:

1. Let $\mathbf{S} = \mathbf{X}$.
2. For each $\mathbf{x}_i$ in $\mathbf{X}$ do:
   - Discard $\mathbf{x}_i$ from $\mathbf{S}$ if it is misclassified using the k-NN rule with prototypes in $\mathbf{X} - \{\mathbf{x}_i\}$.

3.2 Editing with the Nearest Centroid Neighborhood

The nearest centroid neighborhood (NCN) [9] refers to a concept in which neighborhood is defined taking into account not only the proximity of prototypes to a given input sample but also their symmetrical distribution around it. From this general idea, the corresponding classification rule, the k-nearest centroid neighbours (k-NCN) [10], has been proven to overcome the traditional k-NN classifier in many practical situations. Now the NCN editing (NCNE) approach presented here corresponds to a slight modification of the original work of Wilson and basically consists of using the leaving-one-out error estimate with the k-NCN classification rule.

3.3 Proximity Graph Based Editing

Proximity graph based editing scheme are based on the concepts of Gabriel Graph (GG) and Relative Neighborhood Graph (RNG) [11]. The method consists of applying the general idea of Wilson's editing algorithm [5] but using the graph (GG or RNG) neighbors of each sample, instead of the Euclidean distance-based neighborhood, in order to estimate whether a sample is mislabeled or not. In a few words, the simplest GG and RNG prototypes based editing can be summarized as follows: after computing the graph neighborhood of every sample in the original training set, discard those samples that are misclassified by their graph neighbors (instead of their k nearest neighbors).

This editing technique provides some advantages as compared to conventional methods. Firstly, it considers the neighborhood size as a characteristic which depends on each one of the prototypes in the training set. Secondly, GGE and RNGE provides some kind of information about prototypes close enough but homogeneously distributed around a given sample, which can be specially interesting to detect outliers close to the inter-class or decision boundaries. A more detailed description of GGE and RNGE can be found in [12].

4 Experimental Results

The experiments were carried out on ten real data sets taken from the UCI Machine Learning Database Repository (`http://archive.ics.uci.edu/ml/`). A brief summary is given in the Table 1. For each database, we have estimated the overall accuracy by 5–fold cross–validation: each data set was divided into five equal parts, using four folds as the training set and the remaining block as independent test set. The Overall Accuracy were calculated from the equation:

$$Acc = 1 - \frac{n_e}{n_t} \tag{1}$$

where n_e is the number of misclassified examples and n_t is the total number of testing examples.

The experiments have been performed using the Weka Toolkit [13] with the learning algorithms described in Section 2, that is, MLP, SVM and RBF. Each classifier has been applied to the original training set and also to sets that have been preprocessed by the methods NNE, RNGE, NCNE and GGE. These editing approaches has been

Table 1. Data sets used in the experiments

Data set	Classes	Features	Samples training	Samples test
Diabetes	2	8	614	154
German	2	24	800	200
Heart	2	25	217	55
Liver	2	6	276	69
Phoneme	2	5	4323	1081
Sonar	2	60	166	42
Cayo	11	4	4815	1204
Ecoli6[1]	6	7	265	67
Fetwell	5	15	8755	2189
Satimage	6	36	5148	1287

[1] Ecoli6 is obtained from Ecoli. In this work classes 7 and 8 have been eliminated since these only have two samples.

Table 2. Size reduction rate using different editing techniques

Data set	NNE	RNGE	NCNE	GGE
Diabetes	29.75	22.60	32.03	23.08
German	31.71	23.47	33.70	26.46
Heart	35.48	29.03	41.47	31.33
Liver	35.74	29.96	38.98	31.04
Phoneme	11.40	7.83	9.85	12.18
Sonar	18.56	17.96	18.56	34.13
Average	27.11	21.81	29.1	26.37
Cayo	8.20	6.41	9.01	8.20
Ecoli6	14.66	14.28	17.66	18.42
Fetwell	1.77	1.49	1.31	7.38
Satimage	9.53	7.20	9.53	17.98
Average	8.54	7.35	9.38	12.99

improved the classification accuracy of an Nearest Neighbor Rule classifier [14]. Accordingly, this paper addresses the problem of selecting prototypes in order to improve the classification accuracy of an ANN classifier.

The Table 2 reports the percentage of size reduction yielded by the different editing algorithms over ten databases. The results show that in the case of the two-classes data sets, the average reduction rate was approximately 26.1% , meanwhile in the situation of multi-class data sets was 8% to 17% (approximately). From this, the NCNE and the GGE achieve the highest rates in the two-class and multiclass data sets, respectively. Consequently, this implies a important decrease in computational cost in the learning phase. This issue is important in applications with high-size data sets.

On other hand, in Feltwell the number of discards was minimal (except for the GGE method). This is due to the measuring criteria used by NNE, NCNE, and RNGE suggest

Table 3. Experimental results (Acc) for three different artificial neural network architectures

Neural Network	Data set	Original	NNE	RNGE	NCNE	GGE
MLP	Diabetes	75.01(4.88)	74.74(3.56)	75.66(4.31)	75.65(3.05)	75.39(1.69)
	German	70.30(2.20)	72.50(3.81)	73.90(2.38)	72.50(2.57)	71.20(1.48)
	Heart	82.59(4.83)	81.11(7.57)	78.52(4.83)	78.52(4.65)	80.74(4.46)
	Liver	70.14(4.65)	69.28(5.46)	64.06(4.74)	68.12(8.51)	73.04(3.01)
	Phoneme	80.96(1.64)	81.48(1.19)	81.24(1.23)	80.92(1.34)	81.74(0.89)
	Sonar	86.52(5.03)	82.69(1.99)	86.53(2.78)	85.57(5.64)	76.91(2.86)
	Average	77.59(6.8)	76.97(5.55)	76.65(7.61)	76.88(6.20)	76.50(4.17)
	Cayo	86.59(0.29)	86.29(0.34)	86.18(0.33)	86.48(0.59)	86.48(0.71)
	Ecoli6	86.73(2.97)	87.65(3.27)	85.24(2.47)	87.35(1.67)	86.75(1.23)
	Fetwell	95.11(0.77)	95.53(0.48)	94.80(0.37)	95.88(0.44)	94.90(0.27)
	Satimage	88.78(0.67)	89.62(0.49)	89.49(0.66)	89.68(0.58)	87.29(1.56)
	Average	89.30(4.00)	89.77(4.07)	88.93(4.32)	89.85(4.24)	88.86(4.04)
Neural Network	Data set	Original	NNE	RNGE	NCNE	GGE
RBF	Diabetes	72.91(4.93)	75.13(4.76)	74.35(6.67)	72.78(5.55)	75.26(4.65)
	German	74.50(4.11)	70.80(2.46)	72.60(3.27)	72.30(3.83)	71.70(1.96)
	Heart	80.00(6.06)	81.11(5.14)	80.37(4.26)	82.59(4.65)	79.63(3.46)
	Liver	65.22(2.29)	61.45(3.64)	65.80(4.42)	61.45(2.20)	65.22(3.55)
	Phoneme	78.57(1.29)	77.81(1.20)	77.54(0.78)	77.98(1.71)	78.18(0.66)
	Sonar	74.52(8.18)	72.57(6.61)	74.97(5.68)	76.42(4.06)	77.89(6.25)
	Average	74.29(5.20)	73.15(6.81)	74.27(4.96)	73.92(7.17)	74.65(5.40)
	Cayo	87.26(0.58)	87.81(0.38)	88.14(0.54)	87.86(0.55)	87.71(0.47)
	Ecoli6	85.83(2.37)	84.94(2.09)	86.76(2.82)	87.35(2.75)	84.64(1.23)
	Fetwell	90.63(0.61)	89.99(1.72)	89.99(0.95)	89.99(1.25)	91.08(0.76)
	Satimage	84.01(0.48)	85.86(0.83)	86.05(0.99)	85.84(0.86)	85.70(1.12)
	Average	86.93(2.80)	87.15(2.24)	87.74(1.74)	87.76(1.72)	87.28(2.83)
Neural Network	Data set	Original	NNE	RNGE	NCNE	GGE
SVM	Diabetes	76.95(1.97)	75.65(4.15)	76.82(2.53)	76.56(3.01)	76.56(3.17)
	German	75.40(3.31)	72.30(1.89)	73.60(2.13)	74.00(1.90)	71.00(0.71)
	Heart	84.07(4.46)	81.48(6.93)	81.85(4.22)	81.11(5.14)	82.96(3.56)
	Liver	58.55(1.30)	57.97(0.00)	57.97(0.00)	58.26(0.65)	57.97(0.00)
	Phoneme	77.37(1.08)	77.00(1.14)	77.48(1.13)	77.15(1.23)	77.59(1.04)
	Sonar	79.28(7.07)	82.21(4.01)	81.72(2.81)	79.31(4.06)	78.37(5.57)
	Average	75.27(8.72)	74.42(8.90)	74.90(8.90)	74.40(8.27)	74.06(8.80)
	Cayo	66.94(0.48)	66.37(0.13)	66.62(0.48)	65.36(0.39)	69.13(1.00)
	Ecoli6	83.43(1.51)	83.13(1.37)	85.55(0.75)	84.32(3.03)	84.94(1.08)
	Fetwell	91.01(0.14)	91.25(0.23)	91.22(0.26)	91.15(0.28)	91.07(0.33)
	Satimage	86.53(0.99)	86.54(0.84)	86.40(0.86)	86.65(0.85)	85.45(0.99)
	Average	81.98(10.50)	81.82(10.83)	82.45(10.84)	81.87(11.37)	82.65(9.43)

that this data set is not overlapped, i.e., the decision boundary is well defined. However, it is doubtful whether this is really so or if necessary use other measures to locate the prototypes in the overlap region.

Examining the Table 3, the results shown that the editing algorithms obtain similar classification accuracy (Acc) to that of original data sets (i.e., without performing editing). This indicate that, the shrink of training set give a lower computational loads, but also not involve a loss performance. In this sense, although in the case of NCNE no outperform the original data sets (for three classifiers with two-class data set), the differences in such cases are not statistically significant. A final remark from the experiments and perhaps important, refers to that in certain cases the classification accuracy is improved.

In summary, the observations reported in Table 3 suggest that, on multiclass problems the performance editing techniques is better than that of two class problem, and illustrate that these techniques have a tendency to improve the classification accuracy.

5 Conclusion and Further Extensions

When using an ANN, the presence of mislabelled prototypes can degrade the corresponding classification accuracy. Many models for identifying and removing outliers have been proposed. This paper has reviewed some works in the frame of editing the nearest neighbor rule. A number of experiments over ten real data sets have been carried out in order to evaluate the accuracy of those editing methods. The experiments illustrate that editing techniques have a tendency improve the classification accuracy.

In the other hand, the editing techniques here studied, we can diminish the size of the data bases, and with this the diminution of the computational cost and the learning time in the ANN (RBFNN, MLP and SVM). Especially, in two class problem (was reported more of the 21% of reduction, approximately).

Future work is primarily addressed to investigate the potential of these editing methods applied to the hidden space of the neural network. This idea could generate great expectations. In order to obtain this one would be to use a dissimilarity measurement in the transformation space of the training sample and not in the feature space, such as commonly happens with the Wilson editing and its variants.

Acknowledgment

This work has been partially supported by the Spanish Ministry of Science and Education under project CSD2007-00018, UAEMCA-114 and SBI112 from the Mexican SEP, FE28/2009 (103.5/09/4195) of the Mexican PROMEP and the 2703/2008U from the UAEM project.

References

1. Foody, G.: The significance of border training patterns in classification by a feedforward neural network using back propagation learning. JRS 20(18), 3549 (1999)
2. Kavzoglu, T.: Increasing the accuracy of neural network classification using refined training data. Environmental Modelling & Software 24(7), 850–858 (2009)

3. Sánchez, J., Barandela, R., Marqués, A., Alejo, R., Badenas, J.: Analysis of new techniques to obtain quality training sets. Pattern Recognition Letters 24(7), 1015–1022 (2003)
4. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning 38(3), 257–286 (2000)
5. Wilson, D.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man and Cybernetics 2(4), 408–420 (1972)
6. Bishop, C.M.: Neural Networks for Pattern Recognition, 1st edn., January 1996. Oxford University Press, USA (1996)
7. Buhmann, M.D.: Radial basis functions: theory and implementations. Cambridge University Press, United Kingdom (2003)
8. Vapnik, V.: Estimation of Dependences Based on Empirical Data. Springer Series in Statistics (Springer Series in Statistics). Springer, New York (1982)
9. Chaudhuri, B.B.: A new definition of neighborhood of a point in multi-dimensional space. Pattern Recogn. Lett. 17(1), 11–17 (1996)
10. Sánchez, J.S., Pla, F., Ferri, F.J.: On the use of neighbourhood-based non-parametric classifiers. Pattern Recogn. Lett. 18(11-13), 1179–1186 (1997)
11. Jaromczyk, J., Toussaint, G.: Relative neighborhood graphs and their relatives. Proceedings of the IEEE 80(9), 1502–1517 (1992)
12. Sánchez, J.S., Pla, F., Ferri, F.J.: Prototype selection for the nearest neighbour rule through proximity graphs. Pattern Recogn. Lett. 18(6), 507–513 (1997)
13. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann Publishers Inc., San Francisco (2005)
14. Sánchez, J., Barandela, R., Marqués, A., Alejo, R.: Performance evaluation of prototype selection algorithms for nearest neighbor classification. In: SIBGRAPI 2001: Proceedings of the XIV Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2001), Washington, DC, USA, pp. 44–50. IEEE Computer Society, Los Alamitos (2001)

A New Algorithm for Generalized Wavelet Transform

Feng-Qing Han, Li-He Guan, and Zheng-Xia Wang

Chongqing Jiaotong University, 400074 Chongqing, China
hanfengqing@cqut.edu.cn

Abstract. In order to enhance the speed of wavelet transform in signal processing, in this paper an accelerative computing theory is elaborated for generalized wavelet transform and the fast lifted wavelet transform from the perspective of the multi-resolution analysis theory. The capability of accelerative algorithm is proved in theory. Then the accelerative computing procedure for a series of bi-orthogonal Haar wavelet is demonstrated. Appling this idea to multi-resolution representation for medical image, the quality of image is retained and the running time is saved effectively.

Keywords: Accelerative algorithm, Multi-resolution, Embedded sub-analysis.

1 Introduction

A common application of the Discrete Wavelet Transform (DWT) is in image and signal processing[1-3]. Image compression is playing an important role in the modern life with a rapid increase in the amount of digital camera. But in many fields, the DWT is quite a costly operation. So some people proposed fast algorithm to speed it up[4-6].

On the other hand, the generalized fast wavelet transform is a linear time algorithm when the supports of filters are uniformly bounded[7]. But for general filters, the time complexity of wavelet transform algorithm may be nonlinear. So it is necessary to study the time complexity of wavelet transform and the accelerative algorithm when the supports of filters are not uniformly bounded.

In this paper the accelerative computing theory, algorithm for generalized fast wavelet transform is proposed. This accelerative algorithm can be applied into the lifted wavelet transform. The paper is organized as follows. In section 2, the accelerative algorithm for generalized wavelet transform is presented. The author intends to give analysis of time complexity for accelerative algorithm. Section 3 presents a series of bi-orthogonal Haar wavelet and accelerative computing procedure. Section 4 applies this idea to multi-resolution representation for medical image. Finally, section 5 contains the conclusion for this paper.

2 Accelerative Algorithm for Generalized Wavelet Transform

2.1 Accelerative Algorithm Theory

The generalized multi-resolution analysis theory is the foundation for fast wavelet transform. It is defined as follows[7].

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 311–318, 2010.

Definition 1. A multi-resolution analysis M of L_2 is a sequence of closed subspaces $M = \{V_j \subset L_2 \mid j \in J \subset Z\}$, so that

1. $V_j \subset V_{j+1}$,
2. $\bigcup_{j \in J} V_j$ is dense in L_2,
3. For $\forall j \in J, V_j$ has a Riesz basis given by scaling functions $\{\varphi_{j,k} \mid k \in K(j)\}$.

Where $L_2 = L_2(X, \Sigma, \mu)$ is a general function space, with $X \subset R^n$ being the spatial domain, $\sum$ is a σ-algebra, and μ is a non-atomic measure on $\sum$. We do not require the measure to be translation invariant, so weighted measures are allowed.

The $K(j)$ is regarded as general index set and $K(j) \subset K(j+1)$. Let M and M' are two multi-resolution analysis of L_2, where $M' = \{V_{j'} \subset L_2 \mid j' \in J' \subset Z\}$. For describing the procedure of accelerative algorithm theory, we give the following definition.

Definition 2. A multi-resolution analysis M' is called embedded sub-analysis of M if:

1. $M' \subset M$,
2. $\tilde{M}' \subset \tilde{M}$, where $\tilde{M}$ is a dual multi-resolution analysis of M,
3. For $\forall j' \in J'$, $V_{j'}$ has the same scaling functions base and wavelet functions base in M and M', either do $\tilde{V}_{j'}$.

This definition shows that for $\forall f \in L_2, f$ has the same projection coefficients in M and M'.

Assume $V_p \in M, V_q \in M$ and $p > q$.

For $\forall f \in L_2$, the wavelet transform from V_p to V_q is denoted as $Wf(M, p, q)$ in M.

If the V_p and V_q are belong to M', and M' is embedded sub-analysis of M. It is clear that the efficiency of $Wf(M', p, q)$ is higher than one of $Wf(M, p, q)$.

Because of $J \subset Z$, so the index set J is discrete.

Definition 3. For $\forall p, q \in J, p > q$, p and q are called partners in set J, if and only if for $\forall r \in J$, $r \geq p > q$ or $p > q \geq r$.

Specially, there is only one step in the $Wf(M', p, q)$ when the indices p and q are partners in set J'.

2.2 Accelerative Algorithm Procedure

In pseudo code the procedure of $Wf(M, p, q)$ is as follows[7].

```
for j = p-1  downto  q  step -1
   λ_j = H̃_j λ_{j+1}
   γ_j = G̃_j λ_{j+1}
next j
```

When the index p and q are partners, the procedure of $Wf(M', p, q)$ is

$$\lambda_q = \tilde{H}\ \lambda_p\ ,\ \gamma_q = \tilde{G}\ \lambda_p\ , \tag{1}$$

where $\tilde{H} = \tilde{H}_q \tilde{H}_{q+1} \cdots \tilde{H}_{p-1}$, $\tilde{G} = \tilde{G}_q \tilde{H}_{q+1} \cdots \tilde{H}_{p-1}$. It is the accelerative algorithm.

These two algorithms lead to the same results are followed:

$$\lambda_q = \tilde{H}_q \tilde{H}_{q+1} \cdots \tilde{H}_{p-1} \lambda_p\ , \gamma_q = \tilde{G}_q \tilde{H}_{q+1} \cdots \tilde{H}_{p-1} \lambda_p\ . \tag{2}$$

2.3 Analysis of Time Complexity for Accelerative Algorithm

Now we analyze the time complexity of these algorithms. The running time of the algorithm is decided by matrix multiplication, which contains scalar multiplication and addition. The number of scalar multiplication is denoted as $count(M, j)$ In the j-th iteration cycle. So we have:

$$\begin{aligned} count(M, j) &= r(\tilde{H}_j) \times c(\tilde{H}_j) \times c(\lambda_{j+1}) + r(\tilde{G}_j) \times c(\tilde{G}_j) \times c(\lambda_{j+1}) \\ &= r(\tilde{H}_j) \times c(\tilde{H}_j) + r(\tilde{G}_j) \times c(\tilde{G}_j) \end{aligned}\ , \tag{3}$$

where the $r(\cdot)$ and $c(\cdot)$ represent the dimensions of matrix.

Total number of scalar multiplication in $Wf(M, p, q)$ is:

$$count(M) = \sum_{j=p-1}^{q} [r(\tilde{H}_j) \times c(\tilde{H}_j) + r(\tilde{G}_j) \times c(\tilde{G}_j)]\ . \tag{4}$$

From the matrix multiplication and multi-resolution analysis definition, we have the following equations.

$$\forall j \in J,\quad c(\tilde{H}_j) = c(\tilde{G}_j) = r(\lambda_{j+1})\ ,\ r(\tilde{H}_j) + r(\tilde{G}_j) = r(\lambda_{j+1})\ . \tag{5}$$

This results in

$$\begin{aligned} count(M) &= \sum_{j=p-1}^{q} [r(\tilde{H}_j) \times c(\tilde{H}_j) + r(\tilde{G}_j) \times c(\tilde{G}_j)] \\ &= \sum_{j=p-1}^{q} c(\tilde{H}_j)^2 \end{aligned}\ . \tag{6}$$

Similarly, the total number of scalar multiplication in $Wf(M', p, q)$ is:

$$count(M') = r(\tilde{H}) \times c(\tilde{H}) \times c(\lambda_p) + r(\tilde{G}) \times c(\tilde{G}) \times c(\lambda_p) = c(\tilde{H}_{p-1})^2\ . \tag{7}$$

So the efficiency of accelerative algorithm is defined as $1 - (c(\tilde{H}_{p-1})^2 \Big/ \sum_{j=p-1}^{q} c(\tilde{H}_j)^2)$. If $c(\tilde{H}_{k-1}) \big/ c(\tilde{H}_k) = 1/2$ for $\forall k$, The accelerative efficiency are shown in Table 1.

In this case, we have $1-(c(\tilde{H}_{p-1})^2 \Big/ \sum_{j=p-1}^{q} c(\tilde{H}_j)^2) \to 1/3$, when $p-q \to \infty$.

Note that in the accelerative algorithm there is no restriction on the filters. If the support of wavelet filters is uniformly bounded, we only care for scaling coefficients. This conclusion is also correct.

Table 1. Accelerative efficiency with different $p-q$

$p-q$	2	3	4	5
Accelerative efficiency	20.00%	23.81%	24.71%	24.93%

3 Accelerative Algorithm for Bi-orthogonal Haar Wavelet

In [7] the generalized orthogonal Haar wavelet is presented. The index set $L(j,k)$ of this Haar wavelet contains either 2 or 3 elements. Now we generalize a series of Haar wavelet. In which the index set $L(j,k)$ contains t elements, instead of 2 or 3 elements. Then the accelerative computing procedure for Haar wavelet is demonstrated.

3.1 Generalized Bi-orthogonal Haar Wavelet

A set of measurable subsets $\{X_{j,k} \mid j \in J, k \in K(j)\}$ is called a nested set of partitioning[7] if it is a set of partitioning, and for $\forall j, \forall k$, there exists index set $L(j,k)$, so that

$$X_{j,k} = \bigcup_{l \in L(j,k)} X_{j+1,l} \cdot \tag{8}$$

The spaces V_j and $\tilde{V}_j$ are defined as[7]

$$V_j = closspan\{\varphi_{j,k} \mid k \in K(j)\} \subset L_2,\ \tilde{V}_j = closspan\{\tilde{\varphi}_{j,k} \mid k \in K(j)\} \subset L_2, \tag{9}$$

where $\varphi_{j,k} = \chi_{X_{j,k}}$, $\tilde{\varphi}_{j,k} = \chi_{X_{j,k}} / \mu(X_{j,k})$.

In the following part we assume for $\forall j, \forall k$, $|L(j,k)| = t \geq 2$, where t is a constant.

It is easy to proof the space V_j generate a multi-resolution analysis denoted as $M(t)$ of L_2, with the scaling function $\varphi_{j,k}$. The space $\tilde{V}_j$ generate a dual multi-resolution analysis with the dual scaling function $\tilde{\varphi}_{j,k}$.

Because there are t elements in index set $L(j,k)$, let

$$L(j,k) = \{m_1, m_2, m_3, \cdots, m_t\}, \tag{10}$$

where $m_1 = k$.

For every $j \in J$, wavelet ψ_{j,m_i} and dual wavelet $\tilde{\psi}_{j,m_i}$ are defined as

$$\psi_{j,m_i} = \tilde{\psi}_{j,m_i} = \frac{\sum_{s=1}^{i-1} \varphi_{j+1,m_s}}{2\sum_{s=1}^{i-1} \mu(X_{j+1,m_s})} - \frac{\varphi_{j+1,m_i}}{2\mu(X_{j+1,m_i})}, \tag{11}$$

where $i \in \{2,3,\cdots,t-1,t\}$.

From Eq.(9) we have

$$\tilde{V}_j = V_j,\ W_j = \tilde{W}_j, \tag{12}$$

where W_j is the orthogonal complement of V_j in V_{j+1}.

Now we get a series of multi-resolution analyses $M(t)$, $t = 2,3,\cdots$. In each of $M(t)$, the scaling functions are bi-orthogonal and orthogonal. Either do wavelets.

For every $M(t)$, because of $\tilde{\varphi}_{j,k} = \chi_{X_{j,k}} / \mu(X_{j,k})$ and $X_{j,k} = \bigcup_{l \in L(j,k)} X_{j+1,l}$, so the filters are below.

$$\forall l \in L(j,k), \quad \tilde{h}_{j,k,l} = \frac{\mu(X_{j+1,l})}{\mu(X_{j,k})}. \tag{13}$$

From Eq. (11), we have

$$\tilde{g}_{j,m_i,l} = \begin{cases} \dfrac{\mu(X_{j+1,l})}{2\sum_{s=1}^{i-1} \mu(X_{j+1,m_s})}, & l \in \{m_1, m_2, \cdots, m_{i-1}\} \\ 1/2, & l = m_i \end{cases}. \tag{14}$$

3.2 Accelerative Algorithm for Haar Wavelet

From the constructing of $M(t)$, we see that every multi-resolution analysis $M(t)$ comes from a nested set of partitioning $\{X_{j,k} \mid j \in J, k \in K(j)\}$. The nested set of partitioning which generate $M(t)$ is denoted as $\{X_{j,k}^{(t)} \mid j \in J, k \in K(j,t)\}$.

For fixed $t_0 \geq 2$, then there are $t_0 - 1$ Haar wavelet bases in every W_j of $M(t_0)$.

The procedure of $Wf(M(t_0), p, q)$ is given by

```
For  j = p-1  downto  q  step -1
```

$\lambda_j = \tilde{H}_j \lambda_{j+1}$

$\gamma_j = \tilde{G}_j \lambda_{j+1}$

```
next j
```

where the filters $\tilde{H}_j, \tilde{G}_j$ are described in Eq.(13) and (14).

Now for the fixed subspace V_p and V_q in $M(t_0)$, we will construct the embedded sub-analysis $M(t_0^{p-q})$ by the following procedure. So that the subspace V_p and V_q belong to $M(t_0^{p-q})$, where the indices p and q are partners.

The set $\{X_{j,k}^{(t_0)} \mid j \in J, k \in K(j,t_0)\}$ are denoted simply as $\{X_{j,k} \mid j \in J, k \in K(j)\}$, which generates the $M(t_0)$. For $\forall j, \forall k$, the $L(j,k)$ in Eq.(8) contain t_0 elements.

Now we define a new set as follows.

$$\Omega = \{X_{j,k} \mid j = p + (p-q)r, j \in J, r \in Z, k \in K(j)\} . \tag{15}$$

For $\forall j, \forall k$, there exists index set $L'(j,k)$, so that $X_{j,k} = \bigcup_{l \in L'(j,k)} X_{j+p-q,l}$, where the number of elements in $L'(j,k)$ is a fixed natural number t_0^{p-q}.

It is easy to validate the space V_j and $\tilde{V}_j$ satisfy three conditions of definition 1, where $j = p + (p-q)r, j \in J$. So a multi-resolution analysis and its dual multi-resolution analysis are generated by respectively space V_j and $\tilde{V}_j$, denoted as $M(t_0^{p-q})$ and $\tilde{M}(t_0^{p-q})$. Now we reach the following conclusion:

Theorem 1. For $\forall j = p + (p-q)r, j \in J$, the scaling functions which construct V_j in $M(t_0)$ are the same as in $M(t_0^{p-q})$, either do $\tilde{V}_j$.

This theorem shows that for $\forall f \in L_2$, f has the same projection coefficients in $M(t_0)$ and $M(t_0^{p-q})$.

So the accelerative procedure for $Wf(M(t_0), p, q)$ is $Wf(M(t_0^{p-q}), p, q)$ which is the wavelet transform from V_p to V_q in $M(t_0^{p-q})$. The procedure is

$$\lambda_q = \tilde{H}\ \lambda_p ,\ \gamma_q = \tilde{G}\ \lambda_p , \tag{16}$$

where the elements of filter $\tilde{H}$ are below

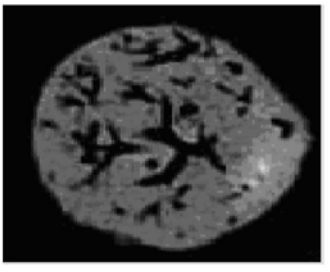
$$\forall l, k, \quad \tilde{h}_{k,l} = \mu(X_{p,l}) / \mu(X_{q,k}) . \tag{17}$$

There are two options of filter $\tilde{G}$. First, dual wavelet bases in $\tilde{M}(t_0^{p-q})$ are the same as in $\tilde{M}(t_0)$, then filter $\tilde{G} = \tilde{G}_q \tilde{H}_{q+1} \cdots \tilde{H}_{p-1}$. Second, dual wavelet bases in $\tilde{M}(t_0^{p-q})$ are constructed with Eq.(11). So the elements of filter are described in Eq.(14) .

4 Experimental Results

In this session we apply the accelerative algorithm in multi-resolution representation for medicine computer tomography (CT) in order to validate its capability. First step is building the partitioning gridding for computer tomography data. Then construct

bi-orthogonal Haar wavelet and accelerate the computing procedure. In this wavelet, we have $|L(j,k)|=2, \text{for } \forall j,k$. Fig.1 illustrates the original medicine CT and transformed image.



a. Original medicine CT

b. Transformed image with $p-q=4$

Fig. 1. Original medicine CT and transformed image

In this experiment the transformed images have the same quality whether the transform is accelerated. This results from theorem 1. But the accelerated transform has saved about 20 percent of time expenditure. This accelerative efficiency is less than theory one in table 1. In fact, besides the wavelet transform the experiment has I/O and other time expenditure.

5 Conclusion

In this paper, a new framework of generalized fast wavelet transform is presented, within which one can accelerate the computing of generalized fast wavelet transform. The analysis of time complexity shows that the running time is saved. This idea can be applicable to the fast lifted wavelet transform and dual lifting scheme as well. The accelerative algorithm is applied to multi-resolution representation for medical image and has a striking effect in running time.

Acknowledgement

This work is supported by NSF Grant #50608072, Natural Science Foundation Project of CQ CSTC 2009BB2375.

References

1. Lee, N., Chambolle, A., DeVore, R., Lucier, B.: Nonlinear Wavelet Image Processing: Variational Problems, Compression, and Noise Removal Through Wavelet Shrinkage. IEEE Trans. Image Proc. 3, 319–333 (1998)
2. Shapiro, J.M.: Embedded Image Coding Using Zerotree of Wavelet Coefficients. IEEE Trans. Signal Proc. 12, 3445–3462 (1993)

3. Amaratunga, K.: A Wavelet-Based Approach for Compressing Kernel Data in Large-Scale Simulations of 3D Integral Problems. IEEE Computing in Science and Engineering 4, 34–45 (2000)
4. Jakob-Chien, R., Alpert, B.K.: A Fast Spherical Filter with Uniform Resolution. J. Comput. Phys. 136, 580–584 (1997)
5. Swarztrauber, P.N., Spotz, W.F.: Generalized Discrete Spherical Harmonic Transforms. J. Comput. Phys. 159, 213–230 (2000)
6. Boulgouris, N.V., Tzovaras, D., Strintzis, M.G.: Lossless Image Compression Based on Optimal Prediction, Adaptive Lifting, and Conditional Arithmetic Coding. IEEE Trans. Image Proc. 1, 1–14 (2001)
7. Sweldens, W.: The Lifting Scheme: A Construction of Second Generation Wavelets. SIAM J. Math. Anal. 2, 511–546 (1997)
8. Daubechies, I., Sweldens, W.: Factoring Wavelet Transforms Into Lifting Steps. J. Fourier Anal. Appl. 3, 245–267 (1998)

Neural Networks Algorithm Based on Factor Analysis

Shifei Ding[1,2], Weikuan Jia[1], Xinzheng Xu[1], and Hong Zhu[1]

[1] School of Computer Science and Technology, China University of Mining and Technology, Xuzhou China 221008
[2] Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing China 100080
dingsf@cumt.edu.cn, dingshifei@sina.com

Abstract. Aimming at the characteristics that the samples to be processed have high-dimension feature variables, and combining with the structure feature of neural networks, a new algorithm of Neural Networkss (NNs) based on Factor Analysis (FA) is proposed. Firstly we reduce the dimensionality of the feature space by FA, regard the data after dimension reduction as the input of the neural networks, and then output the prediction results after training and emulating. This algorithm can simplify the NNs structure, improve the velocity of convergence, and save the running time. Then we apply the new algorithm in the field of pest prediction to emulate. The results show that the prediction precision is not reduced, the error of the prediction value is reduced by using the algorithm here, and the algorithm is effective.

Keywords: neural networks algorithm; factor analysis (FA); feature extraction.

1 Introduction

Neural networkss (NNs) [1] is based on the intelligent computation which uses the computer network system to simulate the biological neural networks. For the NNs algorithm itself, the massive sample features provide available information, but at the same time it also increases the difficulties of processing these data. If it regards all of the data as the inputs, it is not good for the design of the network, and will take a lot of storage space and computing time, too much feature inputs will also cause the time-consuming training process, impede the constringency of the training work, and even ultimately affect the network recognition precision. So it is necessary to process the original data firstly, then analyze and extract useful variable features from the massive data, remove the influence of related or redundant factors, and reduce the dimension of the variable features as much as possible under the premise of not affecting the question solution.

It is also a current hot spot to improve the neural networks through processing the original data. Some scholars proposed some algorithms combined principal component analysis (PCA) and neural networks, such as Lin S K proposed a neural networks prediction algorithm based on clustering and PCA[2]. The proposed algorithm divides samples which have characteristic of decentralization into different sub-classes with

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 319–324, 2010.

the aid of pedigree clustering, and uses principle components analysis method to reduce the dimensionality of the feature space, and then builds neural networks forecasting algorithm. E Lewis et al uses neural networks algorithm based on PCA analyzed the complex optical spectra and time resolution signal[3]. Combined Independent Component Analysis (ICA) and NNs, E S Gopi used ICA to extract the features of forged images, to identify the forged images through NNs [4]. Song JH proposed a new combined algorithm based on ICA and neural networks[5]. Firstly, the independent component analysis algorithm is used for feature extraction. The spectral features are assumed to be a linear mixture of constituent spectra from the material types. The independent components are informative for classification, meanwhile the number of independent components is much smaller than the bands of original data. In addition, some scholars constructed the new neural networks algorithm based on correlation analysis[6], factors analysis (FA)[7], and verified in practice.

However, the research of NNs algorithm based on FA is relatively rare. The original feature variables are processed by standardization and mathematics transformation, it will eliminate the different of distribution indicators and non-comparability caused by numerical difference in order to ensure the data quality. So, through FA, it is not only to avoid information duplicating, but also to overcome the determining weight with the subjective factors. FA concentrates the information of the system's numerous original indexes, it can also adjust the amount of the information by controlling the number of the factors, according to the precision that the actual problems need.

In this paper, firstly, we reduce the dimensionality of the feature space by FA, regard the data after dimension reduction as the input of the neural networks. And then we propose a new algorithm of neural networks based on FA. The purpose is to simplify the NNs structure, improve the velocity of convergence and generalization ability, save the running time, and so it will improve the performance of neural networks. Finally we apply the new algorithm in the field of pest prediction to emulate, and prove that the algorithm is effective.

2 FA-BP Combined Neural Networks Algorithm

2.1 The Basic Principles of FA

FA can be seen as the promotion of PCA, also is a statistical analysis method which changes many feature indexes into several integrated indexes[8,9]. It is a multi-variable statistical analysis method which makes variables with some of the complex relationship transform into the comprehensive factors. It researches on the variables' internal correlated dependencies relationship.

The basic idea of FA is to classify the observation variables, make the variables that are higher correlated in the same class, the correlations among variables of different classes are lower, so in fact each class of variables represents a basic structure, which is public factor. The research problems try to use the sum of the linear function of least number of unobservable public factors and the special factors to describe each component of the original observation. FA has more superiority in the explanation aspect. The basic problem of FA is to determine the load factor by the related coefficient between variables.

FA's main purpose is to simplify the structure of the system. Then we can discover each variable's best subset from numerous factors, describe the multivariable systems results and the influence to the system by the various factor from the information included in the subsets, so as to achieve the goal of dimensionality reduction.

2.2 The Basic Principle of NNs

The NNs is a non-linear, self-adapting information process system which is composed by massive processing units, to process the information by simulating the way of processing and remembering the information by the cerebrum neural networks.

On the network each node can be seen as a neuron, it can memory (storage), process some information, and work with other nodes. When solving the problem, to study some learning guidelines and then simulation output. Based on the powerful fault-tolerance of the NNs, the NNs can easily achieve nonlinear mapping process, and has large-scale computing capacity. Its essence is to gain a parallel and distributed information process functions through the transformation of the network and the dynamics behavior, and simulate the information processing functions of human brain nervous system to some extent, and has some characteristics of self-adapted, self-organization and real-time learning.

At present, hundreds of neural networks algorithms have been proposed, in which Back-Propagation (BP) neural networks [10] is one of most mature and most widespread algorithms, One BP network with three-layer, can approach any continuous function to any precision.

BP algorithm is composed by the forward spread of the data stream and the reverse spread of the error signal. The forward-propagating direction is input layer→ hidden layer→ output layer, the state of neurons of each layer only affect the neurons of next layer, if it can not obtain the expected output, then turns to the process of reverse spread of the error signal. If there are n inputs and m outputs in the network, and there are s neurons in the hidden layer, the output of middle layer is b_j, the unit threshold of middle layer is θ_j, the unit threshold of output layer is θ_k the transfer function of middle layer is f_1, the transfer function of output layer is f_2, the weight from input layer to middle layer is w_{ij}, the weight form middle layer to output layer is w_{jk}, then we can get the output of the network y_k, the desirable output is t_k the output of j th unit of the middle layer is:

$$b_j = f_1\left(\sum_{i=1}^{n} w_{ij}x_i - \theta_j\right) \quad (i = 1,2,\cdots,n\ ;\ j = 1,2,\cdots,s) \tag{3}$$

To calculate the output yk of the output layer through the output of the middle layer,

$$y_k = f_2\left(\sum_{j=1}^{s} w_{jk}b_j - \theta_k\right) \quad (j = 1,2,\cdots,s\ ;\ k = 1,2,\cdots,m) \tag{4}$$

Define the error function by the network actual output,

$$e = \sum_{k=1}^{m} (t_k - y_k)^2 \tag{5}$$

The network training is a continual readjustment process to the weight and the threshold value, to make the network error reduce to a pre-set minimum or stop at a pre-set training step. Then input the forecasting samples to the trained network, and obtain the forecasting result.

2.3 FA-BP Algorithm

Through factor analysis, the dimensionality reduction data of the original data is gotten. Then these low-dimensional data act as the input of the NNs, a new neural networks algorithm based on FA is established, this algorithm can also be referred to as FA-BP algorithm. The basic steps of FA-BP algorithm are as follows:

Step 1. Standardize the original data, and get the correlation coefficient matrix R;

Step 2. Solve the eigenvalue and eigenvector of R, compute the variance contribution rate and the cumulative variance contribution rate, and determine the principal factor;

Step 3. Rotate the factor loading matrix, compute the factor score, and regard it as a new sample;

Step 4. Design BP network structure according to the number of the extracted principal factors and the output of the actual problem, determine the number of hidden layer neurons;

Step 5. Select suitable samples from the new samples as the training samples to train the network, and determine the connection weights and the threshold value;

Step 6. Use the trained FA-BP algorithm to predict the forecasting samples.

3 Case Analysis

Known data[11], predicting the occurrence degree of wheat midge in Guan-zhong area. The forecast of the pest occurrence system can be regarded as an input-output system in essence, the transformational relations include three aspects, that is data fitting, fuzzy conversion and logical-inference, these may be expressed by neural networks.

The meteorological conditions have close relationship with the occurrence of wheat midge, so we use the meteorological factors to forecast the occurrence degree of wheat midge. Here we choose 40 samples from 1961 to 2000 as the research object, use x_1 - x_{14} to express the 14 feature variables (meteorological factor) of the original data which need to be processed, use Y to express the occurrence degree of wheat midge in the same year. Then use standardized methods to standardize the original data (the data after processing are still signed as X).

Using MATLAB toolbox to establish a BP neural networks with 14 neurons in the input layer and 1 neuron in the output layer. In the experiments we choose 30 samples from 1961 to 1990 as the training samples, 10 samples from 1990 to 2000 as the

simulation forecast samples. The results are listed in Table 1 (the error sum of squares is the square sum of the difference of predicted value and actual value).

Using FA-BP algorithm to forecast, carrying out factor analysis for the data which have been standardized with MATLAB software, from this example we extract 5 principal factors. Rotate the factor loading matrix so as to make the factors explain the variables better, and make the factor loading matrix satisfy "the most simple structure criterion", then compute the scores of the factors. After then the algorithm takes the processed data as the inputs of the network, and establish the new BP network for testing.

Table 1. The comparison of the performance of BP algorithm and FA-BP algorithm

NNs Algorithm	Precision	Training Steps	Error Sum Quares
BP Algorithm	80%	799	1.8423
FA-BP Algorithm	80%	406	105875

Table 1 shows that, the comparison of the performance of FABP algorithm and BP algorithm, the forecasting precision of the two new network algorithms has not reduced, but the convergence steps and the error sum of squares actually reduce. Original data through factor analysis get low-dimension data, the network inputs are reduced, it is easy to design the network, the network architecture is simplified, the network training speed is enhanced, and the convergence rate is accelerated. It indicates that the FA-BP algorithms are superior to the traditional BP algorithms, the application is stronger, and worthy popularizing.

4 Results and Discussion

Based on the FA algorithm, combined with BP neural networks algorithm, in this paper we propose FA-BP combination neural networks algorithm. By reducing the feature dimension of original data, the data redundancy is reduced, and also the influence of interrelated, repeated data is excluded. The neural networks has a more powerful ability of processing non-linear problems, the new algorithm that organically combines the two's advantages can better fit more complex non-linear predicted problems. The results of the case analysis show that the FA-BP algorithm has great improvement compare to the single neural networks forecast, the self-learning ability is strengthened, the convergence rate is speeded up, and the run time is saved. Through the data dimension reduction, although certain information has lost, the forecasting precision does not reduce. This article is the main example of predict the pests occurrence used meteorological factors, applications in the future will further, consider the pests occurrence of ecological factors, with a view to improving the Precision of forecasts. The algorithm provides a new and effective way for the study of insect ecology and pest forecast.

Factor analysis which the main purpose is reduced high-dimensional data, when the processing object is a large sample with a large number of data and many feature variables, will better show the advantages. But when the large sample is forecasted by the

neural networks, it is difficult to design the network architecture. To design the network architecture after dimension reduction, so the FA-BP combination neural networks algorithm is more suitable to the large sample with many characteristic variables. During the process of the study, the combination of the multivariate statistical analysis and the neural networks, improves the efficiency of the neural networks in processing problems in great degree. Further extending this thinking, aiming at the characteristics of the forecasting problems, if we choose the appropriate statistical theories, match a ideal neural networks algorithm, then establish a new algorithm, or fix with other intelligent methods, possibly will have new discovery. It is also worth paying attention to such as, improving the dimension reduction algorithms, the improvement of network architecture and the algorithms, the research of the combination way.

Acknowledgements

This work is supported by the Basic Research Program (Natural Science Foundation) of Jiangsu Province of China (No.BK2009093), and the National Natural Science Foundation of China (No.60975039).

References

1. Mccllochw, S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics 5, 115–133 (1943)
2. Lin, S.K., Zhang, D.Y., Li, W.X., et al.: Neural networks forecasting algorithm based on clustering and principle components analysis. Mini-micro Systems 12, 2160–2163 (2005)
3. Lewis, E., Sheridan, C., Farrell, M.O., et al.: Principal component analysis and artificial neural networks based approach to analysing optical fibre sensors signals. Sensors and Actuators A: Physical 1, 28–38 (2007)
4. Gopi, E.S.: Digital image forgery detection using artificial neural networks and independent component analysis. Applied Mathematics and Computation 2, 540–543 (2007)
5. Song, J.H., Feng, Y.: Hyperspectral data classification by independent component analysis and neural networks. Remote Sensing Technology and Application 2, 115–119 (2006)
6. Zheng, F.Q., Sun, C.Z., Lin, J.G.: Study and application on interpretation method of area rainfall based on neural networks. Scientia Meteorologica Sinica 1, 53–57 (2006)
7. Frolov, A.A., Husek, D., Muraviev, I.P., et al.: Boolean factor analysis by attractor neural networks. IEEE Trans. Neural Networks 3, 698–707 (2007)
8. Speraman, C.E.: General intelligence objectively determined and measured. American Journal of Psychology 15, 201–293 (1904)
9. Johnson, R.A., et al.: Applied multivariate statistical analysis, 4th edn. Prentice-Hall, Inc., Englewood Cliffs (1998)
10. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representation by Back-Propagating errors. Nature 6, 533–536 (1986)
11. Zhang, Y.M.: The application of artificial neural networks in the forecasting of wheat midge. Northwest A&F University (2005)

IterativeSOMSO: An Iterative Self-organizing Map for Spatial Outlier Detection

Qiao Cai[1], Haibo He[2], Hong Man[1], and Jianlong Qiu[2,3]

[1] Department of Electrical and Computer Engineering,
Stevens Institute of Technology, Hoboken, NJ 07307, USA
[2] Department of Electrical, Computer, and Biomedical Engineering,
University of Rhode Island, Kingston, RI 02881, USA
[3] School of Science, Linyi Normal University, Linyi 276005, China
{qcai,Hong.Man}@stevens.edu, he@ele.uri.edu, qjllinyi@yahoo.com.cn

Abstract. In this paper, we propose an iterative self-organizing map approach for spatial outlier detection (IterativeSOMSO). IterativeSOMSO method can address high dimensional problems for spatial attributes and accurately detect spatial outliers with irregular features. Detection of spatial outliers facilitates further discovery of spatial distribution and attribute information for data mining problems. The experimental results indicate our proposed approach can be effectively implemented for the large spatial dataset based on U.S. Census Bureau with approving performance.

Keywords: Neural network; Self-organizing map; Mahalanobis distance; Spatial data mining; Spatial outlier.

1 Introduction

Data mining, as a crucial technique in many of today's data intensive applications, aims to extract implicit and useful knowledge from large-scale arbitrary datasets. Among most data mining techniques, the procedure of outlier detection is similarly compared with discovering "nuggets of information" [1] in the large databases. In many situations, the outlier normally carries the important information. However, spatial outlier detection [2] still remains challenging and controversial for several reasons. Firstly, the definition of neighborhood is crucial to determine spatial outliers. Secondly, the statistical approaches for spatial outliers are required to illuminate the distribution of the attribute values for variety of locations compared with the aggregate distribution of attribute values over the all neighboring clusters [3].

In our previous research work, SOMSO [5] [7] was proposed by integrating self-organizing map (SOM) [4] with Mahalanobis distance [6] to detect the spatial outliers. The advantage of SOMSO approach is that it can not only reduce data dimensions, but more importantly, the topological information of spatial location can be preserved to accurately seek similar spatial relationship in large databases. In this paper, we extend the SOMSO approach to be an iterative approach, the

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 325–330, 2010.

IterativeSOMSO method, to improve its efficiency and robustness. The key idea of IterativeSOMSO method is to use the iterative SOM mechanism to effectively determine the neighbor sets to reduce the influence of potential local outliers. Experiment results based on the U.S. Census Bureau database [8] demonstrate the effectiveness of this approach.

The rest of this paper is organized as follows. Section 2 presents the detailed IterativeSOMSO algorithm. In section 3, the detailed simulation analysis of this method is illustrated based on the U.S. Census Bureau databases for spatial outlier detection. Finally, we give a conclusion in section 4.

2 The Proposed Method: IterativeSOMSO

The proposed IterativeSOMSO algorithm can effectively detect spatial outlier with multiple spatial and non-spatial attributes. In this approach, we adopt the Mahalanobis distance concept to determine the threshold for identifying spatial outliers with multiple non-spatial attributes. With iterative utilization of SOM, the neighbor set can be effectively updated to eliminate the influence of potential local outliers for more robust detection.

[IterativeSOMSO Algorithm]

1. Given the spatial dataset $x = \{x_1, x_2, \ldots, x_n\}$ in a space with dimension $p \geq 1$, attribute function f with dimension $q \geq 1$.
2. Normally standardize the non-spatial attribute $f(x)$, i.e., $f(x) \leftarrow \frac{f(x)-\mu_f}{\sigma_f}$.
3. For each spatial point x_i, calculate the neighbor set $N(x_i)$ via SOM as following steps:
   (a) Initialize weight vectors with random small values, for $j = 1, 2, \ldots, N_{neuron}$, where N_{neuron} denotes the number of neurons in the lattice.
   (b) Randomly select a sample from the input data space.
   (c) Search best-matching (winning) neuron at each time iteration through minimum Euclidean distance.
   (d) Update the synaptic weight vector of all neurons.
   (e) Recursively implement step (b) until convergence of the feature map.
4. Compute the neighborhood function $g(x_i)$= average or median of the dataset, and comparison function $h(x_i) = f(x_i) - g(x_i)$.
5. Calculate Mahalanobis distance MD_i as (1).

$$MD_i = \sqrt{(x_i - \mu)^T S^{-1} (x_i - \mu)} \tag{1}$$

   where $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$, $S = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$
6. Search largest Mahalanobis distance and its corresponding spatial index, then remove non-spatial feature of this data item.
7. Iteratively implement Step (4)-(6) until top M outlier candidates emerge.

8. Sort by Mahalanobis distance of the top M outlier candidates in descending order.
9. Let $\chi_q^2(\beta)$ denote chi-square distribution with certain confidence level β, If $MD_i^2 > \chi_q^2(\beta)$, x_i can be identified as a spatial outlier candidate.

The key idea of the proposed IterativeSOMSO algorithm is that distinctive properties of SOM is provided to determine how to organize synaptic weight vectors to represent the original spatial attributes to obtain spatial clusters. SOM can be considered as a special class of artificial neural networks based on competitive learning. The output neuron is essentially the winning neuron placed on the nodes of the lattice (usually one or two dimensions). For various input patterns, the neurons can be selectively adjusted or updated to adapt the competitive learning process. Briefly speaking, the learning SOM involves these stages: competitive phase, cooperative phase and adaptive phase. The principle goal of SOM is to project the input vector with higher dimensions into one or two dimensional discrete map in topologically ordered pattern, which can be effectively used to identify the neighbor set to facilitate spatial outlier detection. In this paper, we also compare the proposed method with the existing technique such as Grid based KNN method [10]. Based on the combination of Grid and KNN method, Grid based KNN approach might improve efficiency to find neighborhood in lower dimension, but it fails to apply this method in spatial outlier detection with high accuracy when spatial attribute dimension increases. A brief description is summarized as follow.

[Grid based KNN method]

1. Construct the specific grid for spatial data.
2. Find the grid index of each spatial point x_i.
3. Those who have the same grid index share the neighborhood relationship.
4. If the number of data points within the same grid is greater than k, search k-nearest neighbors for x_i and then update the neighbor set. Otherwise, keep the results of the neighbor set in step 3.

---

3 Simulation Result Analysis

The "house" dataset, primarily focused on the housing units and building permits in the United States, collects the detailed information about the housing or building ownerships and distribution density. The non-spatial attributes with 5 dimensions include house units in 2000, house units net change percentage from 2000 to 2005, house units per square mile of land area in 2005, housing units in owner-occupied percentage in 2000 and housing units in multi-unit structures percentage in 2000. The experiment shows that SOM approach is provided as an effective tool to detect spatial outliers. It differs from the traditional machine

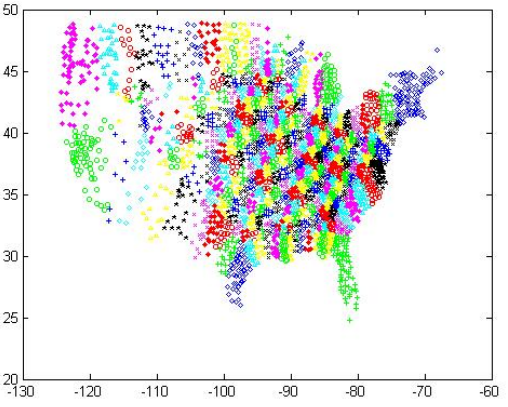


Fig. 1. Spatial clusters: The data points in the clusters with continuously identical marks and colors share common spatial properties in the neighborhood

learning and data mining techniques used in searching spatial neighboring clusters, which are merely concentrated on Euclidean distance for spatial attributes. However, competitive learning promotes synaptic neurons to collaborate with each other and adaptively form feature map with the property of topological ordering. By visual geometric computation in Fig. 1, the proposed method can ultimately acquire important information of the inherent connection for spatial data mining.

Table 1. The top 20 spatial outlier candidates for house dataset detected by Iterative-SOMSO algorithm

Rank	County	Mahalanobis Distance	Units 2000	2000-2005 units net change %	Units per square mile 2005	Units in owner-occupied % in 2000	Units in multi-unit structures % in 2000
1	New York, NY	47.0005	6.8489	0.5159	44.77	7.1244	9.2892
2	Los Angeles, CA	30.1396	29.1008	0.6165	0.8977	3.4559	3.2217
3	Cook, IL	20.1359	18.5294	0.5892	2.7144	2.1363	4.5226
4	Kings, NY	16.6126	8.0432	0.6668	16.7425	6.2006	7.9136
5	Bronx, NY	15.0883	4.0819	0.5752	14.8914	7.1903	8.1802
6	Maricopa, AZ	12.6488	10.9183	2.0930	0.0658	0.8695	1.5582
7	Harris, TX	12.5962	11.3482	1.0502	0.9227	2.4794	2.6778
8	Flagler, FL	11.3892	0.1134	9.0886	0.0347	1.3078	0.2440
9	Hudson, NJ	10.4354	1.8318	0.4953	6.5289	5.7256	7.7003
10	San Francisco, CA	10.3692	2.7849	0.5620	9.4251	5.1582	5.9408
11	Chattahoochee, GA	10.3348	0.3036	0.8650	0.1198	6.2138	1.0996
12	Queens, NY	9.8641	7.0208	0.6698	9.4402	4.1289	6.0901
13	Paulding, GA	9.4858	0.0704	7.2627	0.0390	1.6773	0.7771
14	Suffolk, MA	9.1950	2.2989	0.8840	6.1808	5.3033	7.4231
15	Loudoun, VA	9.0758	0.2259	7.3439	0.0893	0.7008	0.2786
16	King, TX	8.8654	0.3319	0.4893	0.1364	5.2505	1.1610
17	Kenedy, TX	8.6339	0.3309	0.4937	0.1364	5.1846	0.6198
18	Rockwall, TX	8.4423	0.1953	6.3160	0.0792	1.1363	0.2546
19	Dallas, TX	8.2839	7.3522	0.2013	1.1706	2.8357	2.9764
20	Eureka, NV	8.0410	0.3242	0.3352	0.1363	0.0513	0.8944

Table 2. The top 20 spatial outlier candidates for house dataset detected by Grid based KNN algorithm

Rank	County	Mahalanobis Distance	Units 2000	2000-2005 units net change %	Units per square mile 2005	Units in owner-occupied % in 2000	Units in multi-unit structures % in 2000
1	New York, NY	47.1676	6.8489	0.5159	44.77	7.1244	9.2892
2	Los Angeles, CA	33.1943	29.1008	0.6165	0.8977	3.4559	3.2217
3	Cook, IL	19.9999	18.5294	0.5892	2.7144	2.1363	4.5226
4	Kings, NY	17.2194	8.0432	0.6668	16.7425	6.2006	7.9136
5	Bronx, NY	15.6216	4.0819	0.5752	14.8914	7.1903	8.1802
6	Harris, TX	12.4259	11.3482	1.0502	0.9227	2.4794	2.6778
7	Maricopa, AZ	12.3113	10.9183	2.0930	0.0658	0.8695	1.5582
8	Flagler, FL	11.2978	0.1134	9.0886	0.0347	1.3078	0.2440
9	Hudson, NJ	10.8455	1.8318	0.4953	6.5289	5.7256	7.7003
10	Queens, NY	10.6102	7.0208	0.6698	9.4402	4.1289	6.0901
11	San Francisco, CA	10.1129	2.7849	0.5620	9.4251	5.1582	5.9408
12	Chattahoochee, GA	10.0224	0.3036	0.8650	0.1198	6.2138	1.0996
13	San Diego, CA	9.9802	9.0266	0.1974	0.1967	2.4662	2.4646
14	Suffolk, MA	9.9380	2.2989	0.8840	6.1808	5.3033	7.4231
15	Loudoun, VA	9.0355	0.2259	7.3439	0.0893	0.7008	0.2786
16	Henry, GA	9.0182	0.0551	7.2163	0.1149	1.4662	0.3719
17	Paulding, GA	8.9215	0.0704	7.2627	0.0390	1.6773	0.7771
18	Orange, CA	8.8396	8.3907	0.1501	1.4840	1.6744	2.2620
19	Kenedy, TX	8.5330	0.3309	0.4937	0.1364	5.1846	0.6198
20	Alexandria, VA	8.4137	0.2447	0.1052	5.5310	4.4984	5.5250

To better visualize spatial clusters, the topological information will be shown by the cluster density, which can illustrate the number of spatial data items on each neuron. The analysis of cluster density can help us to understand the quantity of spatial data with similar spatial patterns. Besides, the histogram of spatial clusters is employed to display the neighborhood based on the feature map as Fig. 2.

Table 1 and Table 2 illustrate the top 20 ($M = 20$) spatial outlier candidates for house dataset detected by the proposed IterativeSOMSO and the Grid-based KNN algorithm, respectively. From these two tables one can see that both methods can provide comparable results. Since the iterative procedure in IterativeSOMSO can eliminate unknown influence arising from local outliers, we believe

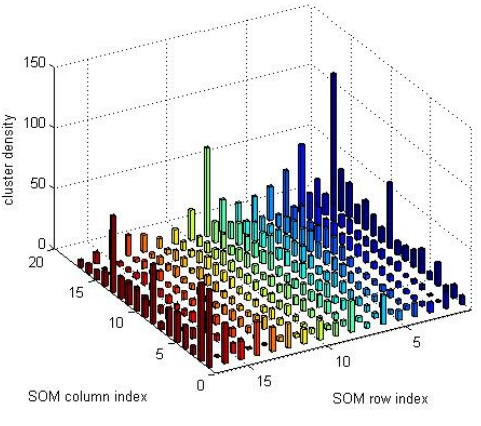


Fig. 2. Histogram of the cluster distribution

the outliers detected by the IterativeSOMSO approach might be more reliable. In terms of computational cost, Grid based KNN will have large computational cost when the dimension of spatial attribute is high. Based on the simulation results, we hope that IterativeSOMSO algorithm may provide an effective approach for such challenging spatial outlier detection applications.

4 Conclusion

In this work, we propose an IterativeSOMSO approach for spatial outlier detection. Experimental results and comparative analysis illustrate the effectiveness of this method. There are a few interesting future directions along this topic. For instance, theoretical analysis of the propose method in terms of convergence is critical to understand the fundamental mechanism of this approach. Also, large-scale experiments and comparative study are necessary to fully justify the effectiveness of this approach. Furthermore, the computational cost of this approach should also be investigated from both a theoretical and empirical point of view. We are currently investigating all these issues and will report their results in the future. Motivated by our results in this paper, we believe the IterativeSOMSO method might be a powerful technique for spatial outlier detection with multiple attributes.

References

1. Larose, D.T.: Discovering Knowledge in Data: An Introduction to Data Mining. John Wiley & Sons Ltd, Chichester (2004)
2. Shekhar, S.: Spatial Databases: A Tour. Prentice-Hall, Englewood Cliffs (2003)
3. Shekhar, S., Zhang, P., Huang, Y., Vatsavai, R.: Trends in spatial data mining. In: Data Mining: Next Generation Challenges and Future Directions, pp. 357–380. AAAI/MIT Press (2003)
4. Kohonen, T.: Self-organizing Maps. Springer, Heidelberg (2001)
5. Cai, Q., He, H., Man, H.: SOMSO: A Self-Organizing Map Approach for Spatial Outlier Detection with Multiple Attributes. In: Proc. Int. Joint Conf. on Neural Networks, pp. 425–431 (2009)
6. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining, pp. 276–277. The MIT Press, Cambridge (2001)
7. Cai, Q., He, H., Cao, Y.: Learning from Spatial Data: A Self-Organizing Map Approach for Spatial Outlier Detection. In: Proc. Int. Conf. on Cognitive and Neural Systems (2009)
8. U.S. Census Bureau, United States Department of Commerce, http://www.census.gov
9. Kohonen, T., Oja, E., Simula, O., Visa, A., Kangas, J.: Engineering Applications of the Self-Organizing Map. Proc. of the IEEE 84, 1358–1384 (1996)
10. Lu, C., Chen, D., Kou, D.: Detecting Spatial Outliers with Multiple Attributes. In: Proc. of 15th IEEE Int. Conf. on Tools with Artificial Intelligence, pp. 122–128 (2003)

A Novel Method of Neural Network Optimized Design Based on Biologic Mechanism

Ding Xiaoling[1], Shen Jin[1], and Fei Luo[2]

[1] School of Computer and Electronic Engineering, Hunan University of Commerce, Changsha 410205, China
lxh_yyy@163.com
[2] School of Computer, South University of Technology, Guangzhou 510092, PRC

Abstract. Optimized design of neural network based on biologic immune modulated symbiotic evolution (BIM) is proposed, which combines with the adjustment of antibody of immune modulated theory so as to keep the individual diversity. With combining evolved intergrowth algorithm and density of immune principle suppress modulation mechanism together, system shortens the individual's length of code and lightened the calculating amount by solving the evolution of the colony to the neuron part, eliminates the premature convergence effectively. Meanwhile, system adopts the improved immune adjustment algorithm, which improved the variety of the colony availably. The neuron that produced in the colony in this way can get and realize the network quickly. The results of simulation experiment which applies in system of the two stands reversing tandem cold mill show that this method is applied to the complicated climate, it has good capabilities of convergence and capability of resisting disturbance.

1 Introduction

Optimization technology is a mathematical basis, used to provide various engineering solutions to the problems of optimizing the application technology, in many areas of the project's quickly spread and applied [1-3]. In engineering, science, economic management and many other fields we frequently encountered the complex optimization problems. When using dynamic programming, branch-and-bound or some other classic algorithms to solve such problems, because the algorithm is too complex to be applied in engineering. And a number of new optimization algorithms, such as Artificial Neural Networks, Chaos[4], Genetic Algorithm, Evolutionary Programming, Simulated Annealing, Tabu Search and Hybrid Optimization Strategy[5], get rapid development through simulating or revealing some natural phenomena or processes, supply new ideas and means to solve complex problems.

Biological immune system is the most complex and the most unique function of the organism. There are many advantages to draw on for the development of new calculation methods. Immunization is a new algorithm based on the principle of immune system, with the multi-peak searching and global optimization capabilities to form multimodal functions. Since the late 20th century, Immune algorithm and its application has become a new field of research and widespread concerned by

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 331–338, 2010.

Computational Intelligence scholars[6], which is another hotspot after the neural networks and evolutionary computation. As the immune algorithm can maintain the diversity of groups adaptively and self-regulatory function, so it has the feature of whole and local search capability features, and is effectively applied at optimization. Immune algorithm has been present in immune function optimization[7], combinatorial optimization, scheduling, network planning[8], neural network optimization and many other optimizations which have applied it and achieved better results.

It means calculating the evolution of the colony that is solved partly that the intergrowth evolves it make up the total solution. And it is solved completely that it carries on operation to the colony with the traditional evolution algorithm to the colony that is solved completely, different from disappearing and solving optimally in the overall situation finally [9]. The intergrowth evolves in a kind of separate search way that runs side by side. Pay attention to the intergrowth among the individuals and cooperation relation further [10], operation is faster in speed. The ones are especially helpful to the solution in complicated environment [11, 12].

On the basis of combining intergrowth evolving with immune adjustment [13], this text proposes Immune Modulated Symbiotic Evolution (BIM) [14, 15], which is used in the neural network. BIM adopts the improved immune algorithm, and it can guarantee the convergence property of the algorithm further.

2 Immune Modulated Symbiotic Evolution

Evolution of the Immune system exists. Through the study we learn the immune system produces antibodies to destroy the invading organism antigen. Immune system contains a large number of lymphocytes, named B cells and T cells which may play a major role in the antibody producing process. When the antigen penetrates into the body, the B cells of high antigen affinity combined with the antigen, splitted in the effect of T cells, then the B-daughter-cells growed in the mother cells on the basis of changes, and the B cells of high antigen affinity survived. This process is called clonal selection. Survivals of B-daughter-cells in clonal selection experience the same process. After several generations, they produced lots of the B cells, which produced antibodies to eliminate antigen. T cells maintain the proper concentration of antibodies by stimulating or inhibiting B cells division.

Through asexual reproduction the immune system achieved that B cells groups evolved to search for the high-affinity B cells which were amplified, and cloned receptor editing was used for local and global search, bone marrow generates new B cells.

In the literature [6], Zuo Xingquan and others brought forward an immune evolutionary algorithm based on the immune system of the evolutionary mechanism. Algorithm defined choice, expansion, replacement and mutant, four species of operation. They introduced the concept of neighborhood, by defining the expanded radius and mutation radius two algorithm parameters construct the smaller neighborhood and the larger neighborhoods, then used these two neighborhoods respectively to give for local and global search expansion and mutation operation, achieved the double-deck neighborhood search mechanism from the overall to the local.

The procedure of BIM Algorithm is as Fig. 1 shows. The Symbiotic Evolution meets one degree of foundations appraised according to neuron behavior of participated in network as there are [16]. It indicates that the neuron with higher adaptability facilitates the form of the optimum network and has better cooperative ability with other neurons at the same time. The immune adjustment that keeps on the basis of variety, even at the advanced evolving stage, can maintain the variety of the colony effectively too. Colony produces in this way at all with neuron whose function shines upon through it may form and realize the solution of the specific task.

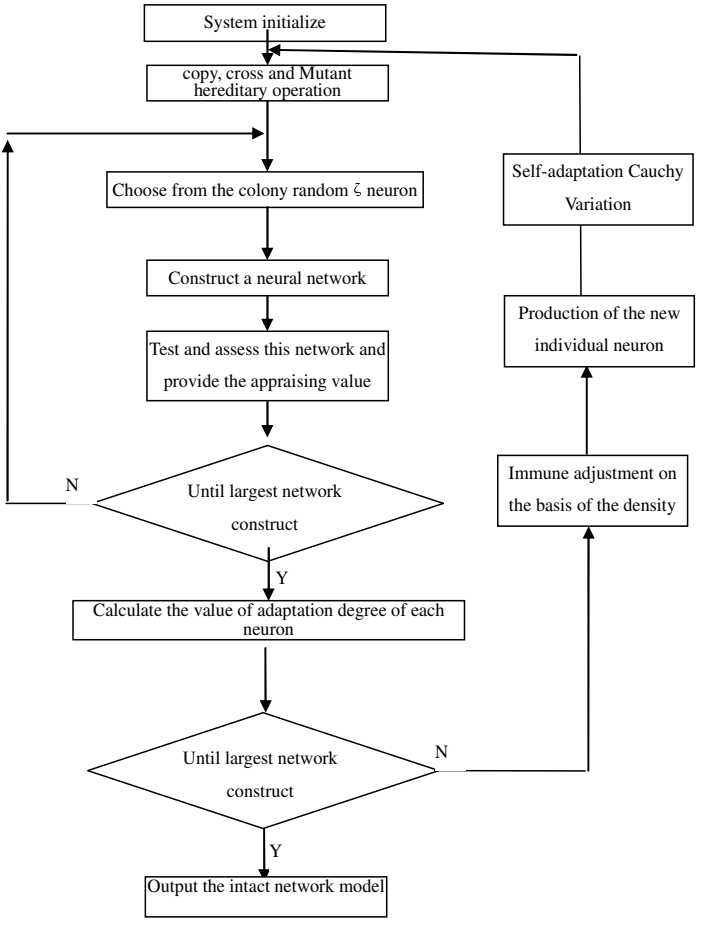


Fig. 1. IMSE arithmetic flow structure

As the individual neuron in the colony is solved partly, we need to choose a neural unit C at random from the colony, construct out the corresponding neural network (namely total solution). In order to adapt the degrees of value to test among environment network by this, we will meet degrees of value which form each adaptation of

neuron of network at the degrees of variable to participate in to get to add, repeat the course of structure of the net and test course, until each neuron has participated in the test of a certain number of times, then individual Ni pieces of adaptation of neuron one degree of value:

$$F(n_i) = \frac{\sum_{k=1}^{\Delta} V(\eta_k)}{\Delta}, n_i \in \eta_k \tag{1}$$

In the type: △—— the times of neuron participating in constructing the network;

V （η） ——network η Adapt ion one degree of appraising value.

Neural network's input node and output node can not be changed which is up to the assignment generally, the neuron that the intergrowth evolves in the colony is used in building the latent layer of nodes. Each neuron should include all the definition of connections with output and input layers. Take a single latent layer of feed forwards network for example, number of connection of each neuron *Num* defines to be:

$$Num = (N+M) \times D \tag{2}$$

In the type: *M*——number of input layer nodes; N——number of output layer nodes; *D*——connect density, generally the fetching value is [0.8, 1]. Each connection includes two pieces of threshold, one is label threshold L_i, the other is power threshold W_i, connecting the chromosome code which forms a neuron with a lot of connections, still includes the threshold value of θ neuron in the code, the neuron individual's code scheme, as the following shows:

L_1	W_1	L_2	W_2	…	L_{Num}	W_{Num}	θ

Label threshold L_i is 8bits unsigned data, representatives connect to specific nodes including input or output *Nod*.

$$Nod(i) = \begin{cases} L_i \bmod(M), & if\ L_i > 127 \\ L_i \bmod(N), & if\ L_i \le 127 \end{cases} \tag{3}$$

i pieces of connections according to grade value of threshold, though it is confirm, it connects to some input or at the node the outputs to calculate. Power threshold is 16 bit signed data, generally assigned as [-5, +5] Real number, whose quantized precision can up to 0.00015 in adjusting power value. All power values of the connection initialize random number among [-1,+1].

As to the same network, if use the standard hereditary algorithm to get the whole neural network code, the number of connections is encoded as $Num=(M+N)\times D\times \xi$ makes one bunch of codes obviously increase, thus it takes long time to search, the calculating amount is heightened, Therefore it's inapplicability to expand to the network with more neurons and joining values. And adopting the method of individual neuron code, the number of units in it increases, the length of code increases with latent layer, and it is easy to carry on the expansion of scale.

Regard the neuron individual as an antibody, utilizing the density of the antibody to choose, the mechanism realizes the function relation that is promoted and suppressed among antibodies, carries on variety to keep improving and has not

disappeared triply, improves the performance that the intergrowth evolves. But if it operates improperly, it will bring unfavorable influence to convergence property of the algorithm. To solve it, improve the realization of immune adjustment algorithm in this text. Narrate it as follows:

(1) **Several concepts**

Information entropy: algorithm for indicating variety of antibody of the colony, introduce information concept of entropy. There is an antibody of N (neuron); the length of code of each antibody is L. It adopts binary scale have the individual's code character collects for {0, 1}, namely character collection of S =2, N piece antibody j information entropy H_j (N) of location it defines to be:

$$H_j(N) = \sum_{i=1}^{S}(-p_{ij}\log p_{ij}) \tag{4}$$

Among them p_{ij} is represent i appear probability at j . Then get average information entropy H (N) of an antibody of N:

$$H(N) = \frac{1}{L}\sum_{j=1}^{L} H_j(N) \tag{5}$$

Degree of kissing among the antibodies: kissing degree between two antibodies u and v of $A_{u,v}$ indicates the similar degree of two antibodies, when it defines to be:

$$A_{u,v} = \frac{1}{1+H(2)} \tag{6}$$

$A_{u,v}$ the fetching value range of v is (0, 1), $A_{u,v}$ is used to show two antibodies kiss or not similar. That $A_{u,v}$ =1 expresses the two code genes are self-same.

The density of the antibody: C_i is antibody density of i defined:

$$C_i = \frac{1}{N}\sum_{w=1}^{N} ac_{iw} \tag{7}$$

Among them: $ac_{iw} = \begin{cases} 1 & ac_{iw} \geq \lambda \\ 0 & otherwise \end{cases}$ λ is a piece of kissing and one degree of constants confirmed in advance, generally the fetching value $0.9 \leq \lambda \leq 1$.

Choose probability: choosing probability to reflect the individual's adaptation degree, the individual's choice probability p is met on $H_j(N) = \sum_{i=1}^{S}(-p_{ij}\log p_{ij})$ e degree of probability p_f and density and suppressed probability p_d two parts to make up. Create definition as follows:

$$p_i = \alpha p_{fi} + (1-\alpha)p_{di} = \alpha\frac{F(i)}{\sum_{j=1}^{N}F(j)} + (1-\alpha)\frac{1}{N}e^{-\frac{c_i}{\beta}} \tag{8}$$

The constants α and β regulate the factor. Type (8) indicates individual meets degrees of F(i) heavy to choose probability to be heavy , but individual density C_i heavy to choose probability to be little.

(2) **Calculation procedure of improved immune adjustment**

STEP1: Initialize: Produce an initial antibody of N at random;

STEP2: Appraise to an antibody of N, Calculate the density of the antibody according to type (7);

STEP3: Calculate the individual's choice probability according to type (8), through choosing the mechanism to carry on the promotion of the antibody and suppress regulating;

STEP4: Upgrade the colony: Use operator of crossing and make a variation operator, cross operator adopt simple single some cross methods, the individual mates at random and cross according to what has been scheduled probability is operated alternately; it adopts adaptive Cauchy function perturbation making a variation method to operate on a variation, this method can survey optimum solution close to the area at present effectively, have certainly gotten rid of some extremely excellent abilities, the concrete method as follows:

Through making a variation probability to choose a variation neuron corresponding weight increases on power value $\triangle x$:

$$W_i=W_i+\triangle x=W_i+T\cdot\tan\ (\pi\cdot R\ (-0.5,\ +0.5)\ ) \tag{9}$$

Among the type R (-0.5, +0.5) is a random variable in the same size among the area [-0.5, +0.5]. The function of density probability of $\triangle x$ is namely that Cauchy is distributed function:

$$f(\Delta x)=\frac{T}{\pi(T^2+(\Delta x)^2)} \tag{10}$$

In the type: T is an adaptive parameter, defined as:

$$T=\frac{T_0}{F_{\max}-F_{avg}} \tag{11}$$

In type: $F_{\max}$——Adaptation degree of the optimum individual in some generation of colony; F_{avg}——Average adaptation degree of the colony; T_0——regulation constant.

The closer between $F_{\max}$ and F_{avg} ,the less difference of adaptation degree among the individuals in the colony, easier to fall into some regions which are extremely excellent. After adding self-adapting parameter, when group tends disappearing too early in the colony, it can change the distribution of Cauchy function, strengthen perturbation value, and regions which are extremely excellent can be surveyed out and the new cyberspace breaks away from them.

3 Examples

Numerical examples from the literature [10]. Done on the basis of the example of the one-year period as follows: planning, the final planning stages load lesser load, take

the time to use the biggest load 5000h, admission 0.35 tariff assuming currency units (Operator cases consistent with the literature [10]). "Group" of the scale of about 10, 100, after the end of evolutionary optimization.

Figure 2 for the same population size, the same evolutionary algebra conditions, Genetic algorithms were applied traditional and new examples of the same genetic algorithm network optimization repeated 50 times the efficiency and resumption Convergence and Stability Results curve. As compared to the same network optimization objective function the same. So two groups of genetic algorithm in the same plane with the evolutionary scale and algebra under the premise optimization process each time for the same cost.

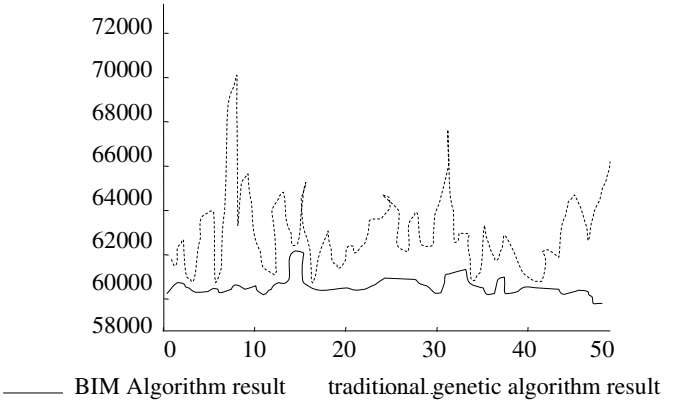


Fig. 2. Computing efficiency and convergence stability comparing curve

The MIMO system can be decomposed into several MISO subsystems to identify, therefore it maintains its universality, so we only discuss the identification of MISO system in the paper.

Can be seen from Figure 1, compared with the traditional genetic algorithm, This paper introduces a new type of genetic algorithm has good stability and convergence of the computational efficiency. Through the optimization have been repeated 50 times the statistical results are as follows:

For example a project of the scale of this optimization problem in optimization hundred behalf of the group of 10 under set conditions. New genetic algorithms and genetic algorithm to find the probability of 0.04 and 0.76 respectively : optimal solution sub-optimal solution within ± 1%. a 0.98 and 0.12 ± 2% probability to find the optimal solution within the scope of sub-optimal solution The probability of finding the optimal solution is closer to 0.54 ± 5% range of sub-optimal solutions.

In addition, due to new genetic algorithms using the "learning paradigm" evolutionary optimization mechanisms, m at the end of the process optimization "paradigm" can provide a different choice of the optimal solution for the planning staff to amend.

4 Conclusions

It has been proved that through the application instance above, on the basis of optimization neural network design method of BIM, with combining evolved intergrowth algorithm and density of immune principle suppress regulation mechanism together, system have shortened the individual's length of code and lightened the calculating amount by solving the evolution of the colony to the neuron part. Meanwhile, system adopted the improved immune adjustment algorithm, which improved the variety of the colony effectively. The neuron that produced in the colony in this way can quickly get and realize the network, which is controlled by the thick and shape of the board.

References

1. Bertotti, G.: Identification of the Damping Coefficient in Landau-Lifshitz Equation. Physical B, 102–105 (2001)
2. Back, T., Schwefel, H.-P.: An Overview of Algorithm for Parameter Optimization. Computation 1(1), 1–23 (1993)
3. Miller, R.K.: 3-D Computing: Modeling, Image Processing, and Noneross. GA: SEAI Visualization Technical (1992)
4. Hearly, G., Binford, T.: IOCAL Shape Form Speculating. Compute. Vis. Graph. Image Process. 42, 62–86 (1988)
5. Dillenbourg, J.: A Self: A Computational Approach to Distributed Cognition. European Journal of Psychology Education 7(4), 252–373 (1992)
6. Maulik, U., Bandyopdhyay, S.: Performance Evaluation of Some Clustering Algorithms and Validity Indices. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1650–1654 (2002)
7. Jiang, W.J., Wang, P.: Research on Distributed Solution and Correspond Consequence of Complex System Based on MAS. Journal of Computer Research and Development 43(9), 1615–1623 (2006)
8. Erg zinger, S., Thomsen, E.: An Accelerated Learning Algorithm for Mrltilayer Percephons: Optimization Layer by Layer. IEEE Trans. Neural Networks 6, 31–42 (1995)
9. He, Y.B., Li, X.Z.: Application of Control Technology Based on Neural Networks. Science Press, Beijing (2000)
10. Jiang, W.J.: Research on the Learning Algorithm of BP Neural Networks Embedded in Evolution Strategies. In: WCICA 2005, pp. 222–227 (2005)
11. Jiang, W.J., Pu, W., Lianmei, Z.: Research on Grid Resource Scheduling Algorithm Based on MAS Cooperative Bidding Game. Chinese Science F 52(8), 1302–1320 (2009)

Research on a Novel Ant Colony Optimization Algorithm

Gang Yi[1,2], Ming Jin[1], and Zhi Zhou[2]

[1] Software School, Hunan University, Changsha, Hunan, 410082, China
6636244@qq.com
[2] Department of Computer, Hunan University of Chinese Medicine, Changsha, China

Abstract. In this paper, an adaptive optimization system is established. In order to improve the global ability of basic ant colony algorithm, a novel ant colony algorithm which is based on adaptively adjusting pheromone decay parameter has been proposed，and it has been proved that for a sufficiently large number of iterations, the probability of finding the global best solution tends to 1. The simulations for TSP problem show that the improved ant colony algorithm can find better routes than basic ant colony algorithm.

1 Introduction

Ant colony algorithm is a new heuristic optimization algorithm initially proposed by Italian scholar Dorigo M, etc in 1991. It adopts positive feedback theory, speeds up evolution course, and it is a parallel algorithm in essence that the individual can cooperate with each other and find better routes by exchanging and transmitting information continuously. Ant colony algorithm is the hot and front research topic in the field of heuristic optimization algorithm both domestic and abroad. Now it has been successfully applied in solving TSP problem, frequency distribution problem in communication, controlling parameter optimization problem and image treatment problem, etc. But at present most scholars only put out algorithm method and application method with no theoretical analysis of algorithm convergence. Until recent years some scholars gave proof of some algorithm convergence. Up to now the most important document about convergence proof is [7-10]. Documents [8, 9] give the graph-based proof of ant colony algorithm convergence based on the theory of describing the pheromone strength on the best routes as Markow stochastic process of discrete time. Document 10 constructs a branch Ant colony algorithm in accordance with stochastic process which proves the algorithm convergence based on the ant quantity, route quantity and birth and death rate by the way of stochastic process. This paper proposes a novel Ant colony algorithm algorithm which is based on adaptively adjusting pheromone decay parameter and proves its convergence by fully considering the effect of the pheromone decay parameter to the algorithm convergence.

2 Basic ANT COLONY ALGORITHM Summarize

Ant colony algorithm is initially applied in solving TSP problem. Let's explain ant colony system by taking TSP problem as example. Suppose: Set c is composed of D

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 339–346, 2010.

cities, m ants, d_{ij} (i,j=1,2,…, D) indicating the distance between City i and City j, l_{ij} (i,j=1,2,…, D) indicating the line between two cities, and $\tau_{ij}(t)$ indicating the remained pheromone strength on route between City i and City j on t hour to simulate the actual ant secretion. Ant k (k=1, 2, … , m) decides its direction according to the information quantity on each route when it moves. And at the same time take notes of the cities Ant k has passed in taboo table $tabu_k$(k=1,2,….,n). The set adjusts dynamically in accordance with evolution process. The ant counts mode shift probability in searching process according to the information quantity and heuristic information on the route. P^k_{ij} indicates the mode shift probability that Ant k shifting from City i to City j on t hour, expressed in formula (1).

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ij}(t)]^{\beta}}{\sum_{s\subset allowed_k}[\tau_{is}(t)]^{\alpha}\cdot[\eta_{is}(t)]^{\beta}}, & If\ \mathrm{j}\in allowed_k\\ 0, & else\end{cases} \tag{1}$$

In this formula, $allowed_k=\{c\text{-}tabu_k\}$ indicates the cities the ants are allowed to choose in next step. α is information heuristic parameter indicating the relative importance of the locus and reflecting the role of the information accumulated in the process of moving. The bigger the value becomes, the more possible the ant chooses the route other ants have passed and the stronger the cooperation among the ants is. β is the expectation heuristic parameter indicating the relative importance of visibility and reflecting the valued degree of the heuristic information in the process of choosing routes. The bigger the value becomes, the closer the mode shift probability is to the greedy rule. $\eta_{ij}(t)$ is the heuristic function. It is expressed in formula (2).

$$\eta_{ij}(t)=\frac{1}{d_{ij}} \tag{2}$$

In order to avoid the remained information covering the heuristic information, renew the remained information after each ant finished one step or passed all D cities. It is expressed in formula (3) and (4).

$$\tau_{ij}(t+D)=(1-\rho)\cdot\tau_{ij}(t)+\Delta\tau_{ij}(t) \tag{3}$$

$$\Delta\tau_{ij}(t)=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t) \tag{4}$$

$\Delta\tau_{ij}(t)$ indicates the pheromone the ants left on the route between City i and City j in this cycle. The algorithm depends on the counting model. In the most used Ant-Cycle model, i.e., formula (5)

$$\Delta\tau_{ij}(t)=\begin{cases}\dfrac{Q}{L_k}, & ant\ k\quad by\ city\ to\quad city\quad j\\ 0, & else\end{cases} \tag{5}$$

Q is the constant indicating the pheromone strength. It affects the convergence speed of algorithm to some degree. L_k is the total length of the route the ant passed in this

cycle. The parameter α,β,ρ in Basic ant colony algorithm, Q may decide its optimum group by experiment. Stopping factors may use fixed evolution algebra or it may stop counting when the evolution tendency is not obvious.

3 The Adoptive ANT COLONY ALGORITHM

Because of the existence of pheromone decay parameter ρ, if ρ is bigger, the information quantity on the unsearched route will reduce to 0 and lower the global searching ability of the algorithm. Although reducing the value of ρ will enhance the global searching ability of algorithm to some degree, the algorithm convergence speed will be lowered. This paper changes the value of ρ by adaptive adjusting according to actual situation. First, give ρ an initial value within (0, 1).

Definition 1
When the better routes remain unchanged after N (N is a constant) times cycle, the better route is doubted to be in partial minimum value.

When the better route is doubted to be in partial minimum, ρ will adjust adaptively by adopting formula (6).

$$\rho(t) = \begin{cases} \xi \cdot \rho(t-1), if & \xi \cdot \rho\ (t-1) \geq \rho_{\min} \\ \rho_{\min}, \quad else \end{cases} \tag{6}$$

In formula (6), ζ(0,1) is decay binding modulus, ρ_{min} is the minimum value of ρ which can prevent ρ lowering the convergence speed when ρ is too small. In addition, compared with basic ANT COLONY ALGORITHM, this algorithm renews the pheromone globally according to the better solution of iterations instead of the partial renewal of pheromone. In this way, the improved algorithm enhances the convergence speed.

Adaptive ant colony algorithm process is as following:

Step 1: Parameter Initialization. Order time t=0, cycle number NC=0. Set the maximum cycle number NC_{max}, randomly distribute m ants to D cities, place the starting city in taboo table $tabu_k$, set the value of α,β,ρ, $\tau_{ij}=\tau_0$, i,j∈R.

Step 2: NC=NC+1;

Step 3: Ant individually chooses City i according to the probabilities counted by mode shift formula (1) and go forward, $j \in \{C - tabu_k\}$

Step 4: Amend taboo table indicator, i.e. remove the ant in a new city after choosing the city and remove the chosen city in the taboo table of the ant individual.

Step 5: If the cities in Set c have not been passed through, i.e. k<m, then k=k+1 and skip to Step3.

Step 6: After m ants finished one cycle, count the route distance L with formula (7) and retain the shortest route L_{min}, order the better route in the first cycle as L_g.

$$L = \sum_{i=1}^{n-1} d\left(c_{tabu_{k(i)}}, c_{tabu_{k(i+1)}}\right) \tag{7}$$

Step 7: Global renewal of pheromone. Renew the pheromone globally according to formula (3),(4) and (8) after the ants passed all cities. If $\tau_{ij}(t+1) \le \tau_{min}$, then order $\tau_{ij}(t+1)=\tau_{min}$.

$$\Delta \tau_{ij}(t) = \begin{cases} \dfrac{Q}{L_{min}}, \\ 0, \end{cases} \tag{8}$$

Step 8: When the better route is doubted to be in partial minimum value, renew the pheromone decay parameter with formula (6).

Step 9: Compared L_{min} with L_g got in this cycle, if $L_{min} < L_g$, order $L_g = L_{min}$ and replace the better route table.

Step 10: If the cycle number $NC \ge NC_{max}$, then the cycle finished and get the better route length L_g and better route table. Otherwise clear away taboo table and skip to Step 2.

4 The Convergence Analysis of New Algorithm

The directions of the principle 1: With regarding to $\forall \tau_{ij}$, there is always having $\tau_{min} \le \tau_{ij} \le \tau_{max}$.Among that, $\tau_{max} \in [\frac{1}{\rho_1} g(s^*), \frac{1}{\rho_{min}} g(s^*)]$.

Proving: Suppose: The optimum route of some on behalf of the ant looking for food is S0, aggregation A={ ρ_1, ρ_2, …, ρ_k } Among that, $\rho_1 \ge \rho_2 \ge \dots \rho_k \ge \rho_{min}$.

(1) The optimum disambiguation does not appear to be like being caught in minimal value of part in t fall generation.

a. If $l_{ij} \notin S_0$,there is $\tau(1)=(1-\rho_1)\tau_0$, $\tau(2)=(1-\rho_1)^2\tau_0, \dots, \tau(n)=(1-\rho)^t\tau_0$.

Think that $t \to \infty$, $\tau(t) \to 0$, it will be forced to arrive at τ_{min}.

b. If $l_{ij} \in S_0$, $\tau(1)=(1-\rho_1)\tau_0+\rho_1\Delta\tau_{ij}$, $g(s^*)=\max(\rho_i \Delta \tau_{ij})$, replace the day after tomorrow by that t falls time, there is $\tau(t)=(1-\rho_1)^n\tau_0+\sum_{i=0}^{t-1}(1-\rho_1)^i g(s^*)$.

There will be $(1-\rho) \subset (0,1)$ because $\rho \subset (0,1)$, so that, $\tau_{max}=\frac{1}{\rho_1} g(s^*)$.

(2) There is k times to be like being caught in minimal value of part time giving a disambiguation preferential treatment most in the t among time of iteration, so ,be based on formula (6), the value of the ρ needs to be adjusted. We write ρ_1 is initial value, ρ_2, ρ_3, …, ρ_n is adjusted value, The corresponding iteration number of times adjusting front is respectively $t_1, t_2, t_3, \dots, t_{k-1}$. $t_k=t, t_0=0$.

a. If $l_{ij} \notin S_0$,

$$\tau(t)=\prod_{i=1}^{k}(1-\rho_i)^{t_i-t_{i-1}}\tau(0) \le (1-\rho_{min})^{t_k}\tau(0)=(1-\rho_{min})^t\tau(0) \quad \text{When} \quad t \to \infty,$$

$\tau(t) \to 0$, it will be forced to arrive at τ_{min}.

b. If $l_{ij} \in S_0$, $M_1 = \sum_{i=0}^{t_1-1}(1-\rho_1)^i$, $M_2 = (1-\rho_2)^{t_2-t_1}\cdot M_1 + \sum_{i=0}^{t_2-t_1-1}(1-\rho_2)^i$

……

$M_k = (1-\rho_k)^{t_k-t_{k-1}}\cdot M_{k-1} + \sum_{i=0}^{t_k-t_{k-1}-1}(1-\rho_k)^i \cdot$

There is $\tau_{ij\max} = \prod_{i=1}^{k}(1-\rho_i)^{t_i-t_{i-1}}\cdot\tau_0 + M_k g(s^*)$, because of the $\rho_1 \geq \rho_i \geq \rho_{\min}$ (i=2,3, …, k). So that,

$$\tau_{ij\max} \geq (1-\rho_1)^{t_k}\tau_0 + \sum_{i=0}^{t_{k-1}}(1-\rho_1)^i g(s^*).$$

$$\tau_{ij\max} \leq (1-\rho_{\min})^{t_k}\tau_0 + \sum_{i=0}^{t_k-1}(1-\rho_{\min})^i g(s^*)\cdot$$

If $t \to \infty$, Method according to above $\frac{g(s^*)}{\rho_1} \leq \tau_{ij\max} \leq \frac{g(s^*)}{\rho_{\min}}$.

Synthetically (1)(2), for $\forall \tau_{ij}$, the $\tau_{\min} \leq \tau_{ij} \leq \tau_{\max}$ is right, among that, $\tau_{\max} \subset [\frac{1}{\rho_1}g(s^*), \frac{1}{\rho_{\min}}g(s^*)]$.[the proving is over]

Theorem 1: Supposing of the P0 (n) is t time of iteration inner algorithm discovering the probability untying S0 giving preferential treatment most first，then with regarding to arbitrarily small $\varepsilon > 0$ and big sufficient iteration number of times t, the $P_0(t) \geq 1-\varepsilon$ is right.

Proving: Because of information amounts τ_{ij} is restricted between $\tau_{\min}$ and $\tau_{\max}$, In the process that ant structure solves, therefore, the feasibility changing probability chooses state as Pmin > 0,what is more, there is

$$P_{\min} \geq \hat{P}_{\min} = \frac{\tau_{\min}^{\alpha}\eta_{\min}^{\beta}}{(D-1)\tau_{\max}^{\alpha}\eta_{\max}^{\beta} + \tau_{\min}^{\alpha}\eta_{\min}^{\beta}}.$$

The letter of D expresses the number of cities in the formula. There will be allowed probability comes into being of the $\hat{P} \geq \hat{P}_{\min}^{n} > 0$ with regarding to S', in the list, n is stand for max. Therefore, $P_0(t) = 1-(1-\hat{P})^t$.Think that, when t is sufficient big, with regarding to arbitrarily small $\varepsilon > 0$,there is $P_0(t) \geq 1-\varepsilon$. [the proving is over]

Theorem 1. gets the probability explaining that the algorithm can be used during the period of the $t \to \infty$,which giving a solution preferential treatment most.

Theorem 2. Set up t * being S0 iteration number of times discover the optimum solution first, $\forall$ (i,j)$\in$ S_0, $\forall$ (k,l)$\in$ L $\wedge$ (k,l)$\notin$ S_0,the there is $\tau_{ij}(t) > \tau_{kl}(t)$ for any t0 when t>t*++t0,among that, $t_0 = \frac{1-\rho_1}{\rho_1}$.

Proving: Consider the very worst condition, order t^* replaces the day after tomorrow w time again and again, there is $\tau_{ij}(t^*) = \tau_{\min}$, $\tau_{kl}(t) = \tau_{\max}$, We in case that t * is at any time corresponding $\rho_1^{'}$, $t^* + t^{'}$ o'clock carve corresponding $\rho_k^{'}$, ($\rho_1 \geq \rho_1^{'} \geq \rho_2^{'} \geq .. \rho_k^{'} \geq \rho_{\min}$), Within the [$t^*$, $t^* + t^{'}$], the optimum disambiguation seems going forward being caught in minimal value of part takes that t0 parts for plain iteration number of times as $t_1^{'}, t_2^{'}, \cdots t_{k-1}^{'}$, we write it as $t_k^{'} = t^* + t^{'}$, In the light of the method ,we can prove that

$$\tau_{ij}(t^* + t^{'}) = \prod_{i=1}^{k}(1-\rho_i^{'})^{t_i^{'} - t_{i-1}^{'}}\tau_{\min} + M_{t^{'}} g(s^*) \quad > \quad M_{t^{'}} g(s^*) \geq t^{'}(1-\rho_1)^{t^{'}-1} g(s^*)$$

$\tau_{kl}(t^* + t^{'}) = \max\{\tau_{\min}, (1-\rho_{\min})^{t^{'}} \cdot \tau_{\max}\}$, $\tau_{ij} \in S_0$, Usually, the overall situation renews ascending, τ_{ij} monotonous ascending, the reason why having $\tau_{ij} > \tau_{\min}$ to be founded according to information. Therefore, if we want to testify $\forall t, \tau_{ij}(t) > \tau_{kl}(t)$ is right, we as lonely as testifying

$$t^{'}(1-\rho_1)^{t^{'}-1} g(s^*) \quad > \quad (1-\rho_{\min})^{t^{'}} \tau_{\max}$$

$$t^{'} > \frac{\tau_{\max} \cdot (\frac{1-\rho_{\min}}{1-\rho_1})^{t^{'}-1} \cdot (1-\rho_{\min})}{g(s^*)} \geq \frac{\frac{g(s^*)}{\rho_1} \cdot (\frac{1-\rho_1}{1-\rho_1})^{t^{'}-1} \cdot (1-\rho_1)}{g(s^*)} = \frac{1-\rho_1}{\rho_1} = t_0 .$$ [the proving is over]

5 Simulation and Discussion

In order to test the efficiency of the improved algorithm, we conduct simulation study to Oliver 30, Ei150 in TSP problem and get the simulation result after 20 times

Table 1. Calculate Result

	Algorithm	Optimum solution	Average solution
Oliver30	Improved ACO	425.29	433.25
	Basic ACO	432.56	440.87
Eil51	Improved ACO	428.39	436.54
	Basic ACO	434.85	440.75

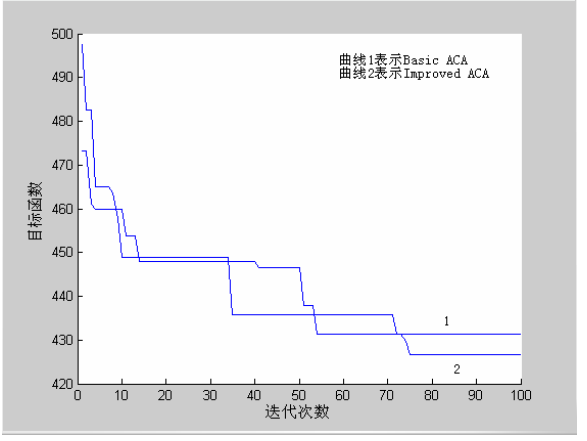


Fig. 1. Oliver 30 the better route evolution curve

operation. The simulation results of different parameter α are in Table 1, its evolution curve in Fig. 1.

By Table 1, we could see that the improved ant colony algorithm has advantage than Basic ant colony algorithm in finding better routes, figure 1 explain this point more clearly. In figure 1, the basic ant colony algorithm will be in partial minimum value in about 55 iterations in solving Oliver30 problem and affect finding the better routes. The improved ant colony algorithm adaptively adjusts the pheromone decay parameter and skips the partial minimum value although its value remains unchanged in 36-71 generations.

6 Conclusions

The pheromone of the basic ant colony algorithm renews through constant decay parameter and this will often cause the algorithm falling in the partial better routes in searching process. This paper enhances the algorithm globalization by adaptively adjusting pheromone decay parameter and avoids falling into the partial minimum value when the ants are searching too concentrated. The theorem in this paper proves the ant colony algorithm convergence and points out that this algorithm can find the global best solution tends to 1 for a sufficiently large number of iterations. Based on the simulation result of TSP problem, the algorithm globalization has been improved and the improved algorithm is effective.

References

1. Coloni, A., Dorigo, M., Maniezzo, V., et al.: Distribution optimization by ant colonies. In: Proceedings of the 1st European Conference on Artificial Life, pp. 134–142 (1991)
2. Dorigo, M., Maniezzo, V., Coloni, A.: Ant system: optimization by a colony of cooperating angents. IEEE Transactions on SMC 26(1), 8–41 (1996)

3. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactional Evolutionary Computing 1(1), 53–56 (1997)
4. Chun-fang, Z., Jun, C., Juan, C.: With finding the solution from the various ant group algorithm fitting in with, frequency assigns problem. The applications of computer 7, 1641–1644 (2005)
5. Hai-bin, D., Dao-bo, W., Xiu-fen, Y.: Novel Approach to Nonlinear PID Parameter Optimization Using Ant Colony Optimization Algorithm. Journal of Bionic Engineering 3, 73–78 (2006)
6. Ge, Y., Meng, Q.C., Yan, C.J., et al.: A hybrid Ant Colony Algorithm for global optimization of continuous multi-extreme function. In: Proceedings if the 2004 International Conference on Machine Learning and Cybernetics, pp. 2427–2432 (2004)
7. Stutzle, T., Dorig, M.: A short Convergence Proof for a Class of Ant Colony Optimization Algorithm. IEEE transom Evolutionary Computation 6(4), 358–365 (2002)
8. Badr, A., Fahmy, A.: A proof of Convergence for Ant Algorithms. Information Science 160, 267–279 (2004)
9. Gutjahr, W.J.: A Graph-based Ant System and Its Convergence. Future Generation Computer Systems 16, 873–888 (2000)
10. Jiang, W.J., Pu, W., Lianmei, Z.: Research on Grid Resource Scheduling Algorithm Based on MAS Cooperative Bidding Game. Chinese Science F 52(8), 1302–1320 (2009)
11. Weijin, J.: Research on the Optimization of the Equipment Fund's Assignment Model Based on HGA. Journal of the Control and Instruments in Chemical Industry 31(2), 10–14 (2004)
12. Jiang, W.J., Wang, P.: Research on Distributed Solution and Correspond Consequence of Complex System Based on MAS. Journal of Computer Research and Development 43(9), 1615–1623 (2006)

A Sparse Infrastructure of Wavelet Network for Nonparametric Regression

Jun Zhang[1,2], Zhenghui Gu[1], Yuanqing Li[1], and Xieping Gao[3]

[1] Center for Brain-Computer Interface and Brain Information Processing
South China University of Technology, Guangzhou, 510640, China
[2] Information Engineering College
Guangdong University of Technology, Guangzhou, 510006, China
[3] Information Engineering College, Xiangtan University, Xiangtan, 411105, China

Abstract. In this paper, we propose a novel 4-layer infrastructure of wavelet network. It differs from the commonly used 3-layer wavelet networks in adaptive selection of wavelet neurons based on the input information. As a result, it not only alleviates widespread structural redundancy, but can also control the scale of problem solution to a certain extent. Based on this architecture, we build a new type of wavelet network for function learning. The experimental results demonstrate that our model is remarkably superior to two well-established 3-layer wavelet networks in terms of both speed and accuracy. Another comparison to Huang's real-time neural network shows that, at similar speed, our model achieves improvement in generalization performance. *abstract* environment.

Keywords: Wavelet network, neural network, sparse infrastructure, Regression.

1 Introduction

As is well known, neural network that has widely used iterative searching methodology in the learning algorithms is marked with the shortcoming of slow convergency speed, particularly in the case of large-scale observations and/or high accuracy expectation .Hence, the real-time handling capacity is a great challenge as far as neural network is concerned. Recently,wavelet was introduced into neural network to take the advantages of both the time-frequency property of wavelets and the effective learning mechanism of neural networks [2,4]. The resultant wavelet neural network has become a powerful tool for many applications. Typically, regarding function learning as a fundamental and vital problem in many fields, Jun Zhang built an orthogonal wavelet network in order to solve the redundancy and non-orthogonal problems of basis in Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) neural network. It ultimately leads the learning problem of model to solution to a set of linear equations. Therefore the iterative searching methodology can be avoided in the construction of model.

However, there still exist at least two problems as far as Jun Zhang's work is concerned. Firstly, in [3], Jun Zhang pointed out that the solution of a set

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 347–354, 2010.

of linear equations may be computation intensive when the scale of problem gets larger. For real-time applications, computation load has to be controlled in solving the linear equations. Secondly, with respect to the whole function, we can build the orthogonal wavelet network for approximation that does not have any redundancy, just as [3] has done. Furthermore, aiming at a specific problem, we can employ certain algorithms for structural adaptation of the wavelet network in order to eliminate unnecessary wavelons. Due to the sparseness of large-dimensional training data, more specifically, wavelets whose supports do not contain any data point can be eliminated in the training process [5]. But concerning part of the function, the wavelet networks built by the above methods still have structural redundancy. That is to say, not all the wavelons are needed to handle part of the function.

Wavelet network model successfully maintains the favorable advantages of the RBF neural network structure. Therefore most scholars are concentrating on the algorithm improvement and pushing forward the study of the wavelet network theory, yet few pay adequate attention to the structural redundancy. For the first question mentioned above, more and more algorithms for wavelet networks have been proposed. While for the second question generally existing in all kinds of wavelet network models, restricted by the present structure of wavelet networks, such redundancy is unavoidable.

Inspired by the idea of Huang et al [6] about the neuron quantizer for general neural networks, in this paper, we propose a novel 4-layer wavelet network architecture for function learning that substantially differs from the previous wavelet network models adopting the RBF architecture. Based on the input, this novel architecture can adaptively select a subset of the wavelons by designing a group of wavelon selectors. Under this architecture, not only the structural redundancy widely existing in wavelet networks can be avoided as much as possible, but it can also control the number of wavelons in each calculation with a view of reducing the scale of the problem solutions. Furthermore, based on this infrastructure, we build a new wavelet networks and evaluates them with function learning. Performance comparison has been made on its generalization ability and learning speed with the aforementioned existing models in [2,4].

2 Orthogonal 3-Layer Wavelet Network Infrastructure

2.1 Wavelet Network for Function Learning

The wavelet decomposition of a function $f(x)$ can be expressed as

$$f(x) = \sum_{n} \langle f, \varphi_{m_0,n} \rangle \varphi_{m_0,n}(x) + \sum_{m \geq m_0} \sum_{n} \langle f, \psi_{m,n} \rangle \psi_{m,n}(x) \tag{1}$$

where $\varphi(\cdot)$ and $\psi(\cdot)$ denotes the scaling function and the wavelet function respectively; $\langle \cdot, \cdot \rangle$ represents the inner product and m_0 is an arbitrary integer representing the lowest resolution or scale in the decomposition. Given a set of training data $T_N = (x_i, f(x_i))_{i=1}^{N}$, the purpose of function learning is to find

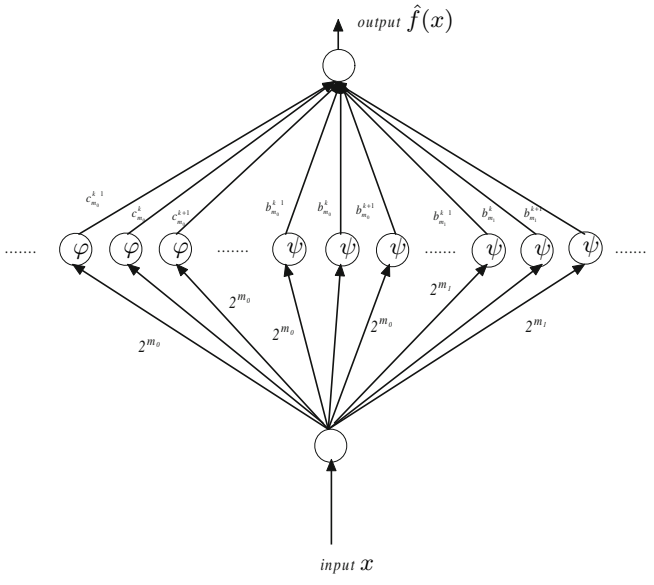


Fig. 1. The 3-layer wavelet network [5]

an estimate $\hat{f}(x)$ that closely approximates $f(x)$. For this purpose, a 3-layer wavelet network model according to the wavelet theory can be built, as shown in Fig.1. The hidden layer contains two kinds of nodes that we call φ set and ψ set both containing countable numbers of nodes. All of these nodes are named as wavelons in this paper. Lemma 2.1 gives the bias of each φ neuron and the size of φ subset. The result is the same with ψ subset.

Lemma 2.1[5]. Without loss of generality, if the support of $\varphi(x)$ is [0,μ] and that of $f(x)$ is [0,1], for the purpose of making $\{\varphi_{m_0,n} = 2^{\frac{m_0}{2}}\varphi(2^{m_0}x - n)\}$ overlay $f(x)$, the most neurons of φ set is $(2^{m_0} + \mu - 1)$ and the corresponding biases are $b_{m_0} = \{-\mu + 1, ..., 2^{m_0} - 1\}$. m_0 represents the lowest resolution or scale in the decomposition.

As a matter of fact, the problem solution can be converted to solution of a set of linear equations which has been discussed in [3]. When k is large, a direct solution to (1), which involves the inversion of a large matrix, may be computation intensive.

2.2 Shortcoming of 3-Layer Wavelet Network

As discussed above and Lemma 2.1, in the practical implementation of the wavelet network, the number of hidden layer nodes is determined by integers m_0, m_1, as well as the supports of both function f and scaling function $\varphi(\cdot)$(or wavelet function $\psi(\cdot)$). For example, suppose the interval $[0, \alpha]$ is the support of the function f and the support of scaling function is [0,μ], then the

number of hidden nodes is $(2^{m_0}\alpha + \mu - 1)$ and the corresponding biases are $b_{m_0} = \{-\mu + 1, ..., 2^{m_0}\alpha - 1\}$. However, for the part of the function f whose interval is $[c, d]$,$(c > 0, d < \alpha)$, the wavelet network only need the hidden nodes whose corresponding biases are $b'_{m_0} = \{2^{m_0}c - \mu + 1, ..., 2^{m_0}d - 1\}$. So, the number of hidden nodes is $(2^{m_0}d - 1) - (2^{m_0}c - \mu + 1) + 1$. In other words, it does not entail so many hidden nodes as far as part of the function is concerned.

Without loss of generality, for d dimensional function f and distinct samples $T_N = (x_i, f(x_i))_{i=1}^N$, where $x_i = [x_{i1}, x_{i2}, ..., x_{id}]^T \in R^d$ and $f(x_i) \in R$, we suppose the support of f is $[0, \alpha]^d$. As far as j-th dimension is concerned, the sample set can be divided into L group as follows.

$$V_j^q = \{x_{ij} \mid \frac{(q-1)\cdot\alpha}{L} < x_{ij} < \frac{q\cdot\alpha}{L}\}, \\ 1 \le q \le L, 1 \le j \le d, q, j \in Z \tag{2}$$

and L can be set as any positive integer. Furthermore, regarding all the dimensions, we can divide the sample set into L^d groups as follows.

$$GroupV^p = \{x_i \mid \frac{(q_j-1)\cdot\alpha}{L} < x_{ij} < \frac{q_j\cdot\alpha}{L}, 1 \le j \le d, \\ j \in Z\}, 1 \le p \le L^d, p = \sum_{j=1}^{d}(q_j - 1)\cdot L^{(j-1)} + 1 \tag{3}$$

We define a partition of function and its sub-functions as follows.

Definition 1: In view of d dimensional function f, a partition of function f is given by $f = f_1 \oplus f_2 \oplus ... \oplus f_{L^d}$, where f_i represent the i-th sub-function of f according to the supports being continuously and equally divided in all the dimensions.

Remark: According to the above discussion, we know that the data belonging to $GroupV^p$ are sampled from sub-function f_p.

Therefore, if a model can adaptively select the necessary hidden nodes to approximate the sub-function according to the input information, then the redundant wavelons for the sub-function can be eliminated accordingly. In the following section, we will introduce a novel wavelet network architecture for this purpose.

3 Proposed 4-Layer Wavelet Network Infrastructure

Assuming x as a d dimensional input, we construct the novel 4-layer wavelet network with two hidden layers and d additional wavelon selection units in the first hidden layer. As shown in Fig.3, each selection unit consists of two neurons named A-type neuron and B-type neuron. The neurons in the j-th selection unit are denoted by A_j and B_j. Regarding an arbitrary input $x \in [0, 1]$, wavelon selection unit plays the role of correctly choosing one of the L^d outputs of the second hidden layer as the final output. Such functionality is realized through choosing the weights and biases of the wavelet selection units as follows.

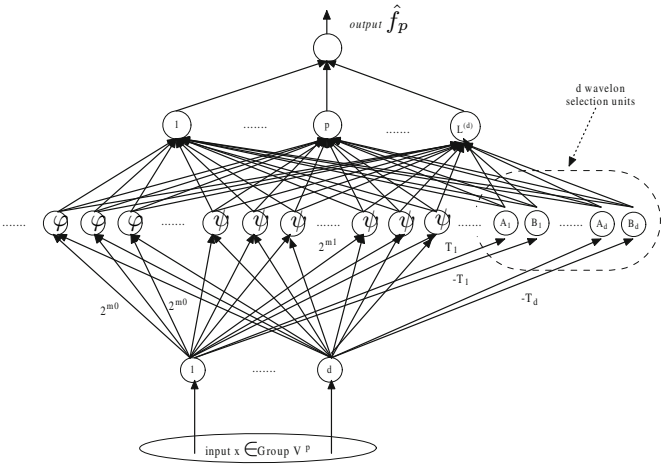


Fig. 2. Four layer architecture of wavelet network

1) The biases $b^q_{A_j}$ and $b^q_{B_j}$ of the two neurons in the j-th wavelon selection unit for the p-th neuron in the second hidden layer, where $1 \le p \le L^d, j = 1, ..., d.$, is designed as follows.

$$\begin{cases} b^q_{A_j} = -T_j(\dfrac{q \cdot \alpha}{L}) \\ b^q_{B_j} = T_j(\dfrac{(q-1) \cdot \alpha}{L}) \end{cases} \tag{4}$$

where $q = \begin{cases} \dfrac{p \mod L^{d-j+1} - p \mod L^{d-j}}{L^{d-j}} & j = d \\ \dfrac{p \mod L^{d-j+1} - p \mod L^{d-j}}{L^{d-j}} + 1 & j \neq d \end{cases}$ $1 \le j \le d \quad 1 \le q \le L.$

2) Denote ω_{A_j} and ω_{B_j} as the weights that link the j-th neuron in the input layer to the j-th wavelon selection unit. Suppose $\omega_{A_j} = -\omega_{B_j} = T_j$,$1 \le j \le d$. Lemma 3.1 and Theorem 3.1 can be given as follows.

Lemma 3.1: Given arbitrarily small positive value $\eta < 0.5$, there exists a constant set

$$Tset = \{T_j^{const} | T_j^{const} = \frac{\ln(\frac{1-\eta}{\eta})}{\min_{\substack{1 \le i \le N \\ 1 \le q \le L}} (x_{ij} - \frac{q \cdot \alpha}{L})}, 1 \le j \le d\}.$$

When $T_j \ge T_j^{const}$, for $\forall x_{ij} \in V_j^k, k = 1, ..., L$, the corresponding output $Output_{A_j^q}$ of neuron A_j for the p-th neuron of the second hidden layer satisfies

$$Output_{A_j^q}(x_{ij}) = \begin{cases} \le \eta, & if\ q \le k \\ \ge 1 - \eta, & if\ q < k \end{cases}$$

Theorem 3.1[1]**:** Give arbitrarily small positive value $\eta < 0.5$, there exists a set $Tset$

$$Tset = \{T_j^{const} | T_j^{const} = \frac{\ln(\frac{1-\eta}{\eta})}{\min\limits_{\substack{1\le i\le n \\ 1\le q\le L}} (\left| x_{i,j} - \frac{q\cdot\alpha}{L} \right|)}, 1 \le j \le d\}$$

when $T_j \ge T_j^{const}$,$1 \le j \le d$,for $\forall x_i \in GroupV^p, p = 1, ..., L^d$, the corresponding $Output^p_{(A_j+B_j)}(x_i)$, satisfies

$$Output^p_{(A_j+B_j)}(x_i) = \begin{cases} \le 2\eta, & x_{ij} \in V_j^q \\ \ge 1 - 2\eta, & x_{ij} \notin V_j^q \end{cases} \tag{5}$$

According to **Theorem 3.1**, $\forall x_i \in GroupV^q$,$q = 1, ..., L^d$, we can get $\eta \to 0$, thus making the inputs of the $q-th$ neuron in the second hidden layer, which come from the all of the wavelon selectors next to zero and the other neurons in the second hidden layer at least have one input that come from the wavelon selectors next to one.According to the orthogonality of wavelet, with respect to each neuron in the second hidden layer, the weights linking it to wavelons from different sub sets of φ set and ψ set can be determinated independently and successively through a transformation to the solution of a set of linear equations. The major advantage of this novel architecture is that the scale of each set of linear equations can be well controlled through function partition. On one hand, it reduces the structural redundancy as much as possibly to improve the generalization performance. On the other hand, the aim of controlling the model's computation load is achievable.

4 Experimental Results

In this section, a four-layer wavelet network (FWN) is constructed based on the novel architecture.To construct the model, we can divide the data set into two subsets, including training set $TraSet_N = (t_i, f(t_i))_{i=1}^N$ and test set $TesSet_M = (t_i, f(t_i))_{i=1}^M$. In order to evaluate the proposed architecture of wavelet networks for function learning, two examples shown in Fig.3 were given to demonstrate the validity of the presented FWN. All simulations of FWN were run in matlab 6.5 environment on an ordinary PC with Pentium 1.7 GHZ processor and 256MB memory. The measure used in [3] was taken as the performance index.

$$Error = \sqrt{\sum_{i=1}^{n} [\hat{f}(t_i) - f(t_i)]^2 / \sum_{i=1}^{n} [\bar{f} - f(t_i)]^2} \tag{6}$$

where $\bar{f} = \frac{1}{n}\sum\limits_{i=1}^{n} f(t_i)$, and n is the number of sampled data. In Table 1, we compare the performance of our FWN model with the existing well established work in [2] [4] on the learning speed and generalization ability with the measure defined in (6). Obviously, both the two performance indexes of our model

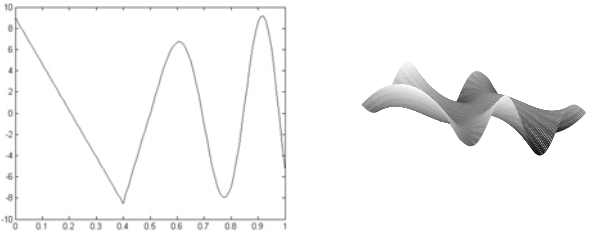


Fig. 3. Function 1 and Function 2 as examples

are remarkably superior to the previous works. Although the number of nodes has been increased, the weights and bias of most nodes can be determined at first. Because the learning algorithms of our models are not gradient-based, the approximation can be performed in a single "epoch".

Table 1. Performance comparison

Function	Model	Hid-Nodes	Epochs	Test-Err	Times(s)
Function 1	Zhang[2]	7	10000	0.0506	1100
	Pati[4]	31	800	0.0240	101.7
	BP	7	10000	0.1329	1150
	FLWN	34	1	0.0024	0.172
Function 2	Zhang[2]	49	40000	0.0339	21300
	Pati[4]	187	500	0.0230	500
	BP	225	40000	0.2938	95640
	FLWN	217	1	0.0085	3.2180

5 Conclusion

In this paper, we proposed a novel 4-layer architecture of wavelet network. Through designing a group of wavelon selection units, it adaptively selects the necessary wavelons based on the input information. The infrastructure can overcome the structural redundancy often met in conventional 3-layer architecture and bring the computation load into the effective control. Hence, it not only effectively improves the generalization performance of the model but also has faster learning capacity than the 3-layer architecture. Based on the experiment results for function learning, compared with the existing wavelet networks, the learning speed and the generalization performance of the new wavelet network models have been greatly improved.

Acknowledgments. This work was supported by the National Natural Science Funds for Distinguished Young Scholar (Grant No.60825306), Natural Science Foundation of Guangdong Province (Key Program) (Grant No.9251064101000012) and Excellent Youth Development Project of Universities in Guangdong Province (Grant No.LYM08050).

References

1. Zhang, J., Gao, X.: A new architecture of wavelet networks for function learning. IET Signal Processing (2009) (submitted to)
2. Zhang, Q., Benveniste, A.: Wavelet network. IEEE Trans. Neural Network 3, 889–898 (1992)
3. Zhang, J., et al.: Wavelet neural networks for function learning. IEEE Trans. Signal Process. 53, 1485–1497 (1995)
4. Pati, Y.C., Krishnaprasad, P.S.: Analysis and synthesis of feed-forward neural networks using discrete affine wavelet transformations. IEEE Trans. Neural Networks 4, 73–85 (1993)
5. Gao, X.P., Zhang, B.: Interval-wavelets neural networks (1)–theory and implements. Journal of software 9, 217–221 (1998)
6. Huang, G.-B., Zhu, Q.-Y., Siew, C.-K.: Real-Time Learning Capability of Neural Networks. IEEE Trans. Neural Networks 17, 863–878 (2006)

Information Distances over Clusters

Maxime Houllier and Yuan Luo

Computer Science and Engineering Department, Shanghai Jiao Tong University, The MOE-Microsoft Key Laboratory for Intelligence Computing and Intelligence Systems, Shanghai 200240, China
{houllier,yuanluo}@sjtu.edu.cn

Abstract. As the data bases grow bigger, the data mining becomes more and more important. In order to explore and understand the data, the help of computers and data mining methods, like clustering, is a necessity. This paper introduces some information theory based distances and how they can be used for clustering. More precisely, we want to classify a finite set of discrete random variables. This classification has to be based on the correlation between these random variables. In the design of a clustering system, the choice of the notion of distance is crucial, some information distances and classification methods are provided. We also show that in order to have distances over clusters, the variables that are functions of other variables have to be removed from the starting set. The last part gives some applications run on Matlab.

Keywords: Information distance, Hierarchical classification, Markov chain.

1 Introduction

A clustering algorithm classifies a set of elements into subsets (called clusters). In this paper, we suppose that our elements are discrete random variables (definition and more information in [1]): this is very practical because nearly everything can be represented by a random variable. The elements that will be set in a same cluster during the classification have to be similar in some sense. This sense is defined by a specific measure (or distance) that has to be chosen before the classification. We are trying to find which random variables work with which other ones, such that, in the final classification, as much as possible, the correlated variables should be classified in the same clusters, and the independent ones should not be together. For that we need a distance that measure the inverse of correlation between the random variables. A high correlation degree will have a small measure, and the more independent the variables are, the greater the distance should be.

Classification of random variables based on correlation has already be treated by scientists. The most usual methods use linear correlation based distances for the classification, like in [2] and [3], but it has been shown in [4] that the mutual information can also be used and seems better. Closer to what we are going to study in these papers: [5] and [6] work on the same problem, they

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 355–364, 2010.

use information distances for clustering of random variables. This paper tries to continue this work.

If two random variables have a bijection between them, then they are completely correlated: they express twice the same information. In information theory, this two variables are considered to be equivalent, we say that they are equal.

Definition 1. *Let X and Y be two random variables. We say that X is equal to Y ($X \sim Y$) if:*

$$\exists \text{ a bijection } f \text{ such that } f(X) = Y.$$

Notation: Because of this definition, we will use $X_{[1,n]}$ to express the random vector: $(X_1, \ldots, X_n)$, where $[1, n] = \{1, \ldots, n\}$. It may seem strange because for a random vector the order of the coordinates is important and in this new notation $X_{[1,n]}$ there is no such an order. But, after noticing that for a random vector and any other random vector that is a permutation of the coordinates of the first random vector, these two random vectors are equal (a permutation of the coordinate is a bijection). Then, with this new equality, the order of the coordinates is not important. We extend this notation to any subset G of $[1, n]$, X_G represents the random vector:$(X_i)_{i \in G}$.

This paper, first, focuses on information theory: entropy and mutual information. In the thrid part, some information theory based distances over random variables are studied. With these tools, the fourth part continues with clustering: how to extend these distances over clusters and how to achieve classification. This paper ends with some tests run on Matlab.

2 Information Theory: Entropy and Mutual Information

The theory of information has first be introduced by Shannon [7], and it has today a lot of applications. It gives to scientists a set of tools that help to measure the quantity of information in the outcome of a random experiment, or passing through a channel. It can also, as we will see with more details later, help to measure the degree of association between two or more events. This part reminds basic definitions of some information theory tools such as the entropy and the mutual information. Let χ (resp. Ψ, Ξ) be the alphabet of the random variable X (resp. Y, Z).The entropy of the random variable X is a real value that represents how much unknown there is in one outcome of X:

$$H(X) = -\sum_{i \in \chi} p_i \log(p_i). \tag{1}$$

If we have two or more variables, we can study the entropy the union of these variables. For example, with three variables, we have:

$$H(X, Y, Z) = -\sum_{(i,j,k) \in (\chi \times \Psi \times \Xi)} p_{i,j,k} \log(p_{i,j,k}). \tag{2}$$

The conditional entropy is a real value that represents how much unknown remains in X if we already know Y, it is defined as:

$$H(X \mid Y) = - \sum_{(i,j)\in(\chi\times\Psi)} p_{i,j} \log(p_{i|j}). \tag{3}$$

The mutual information is an interesting object, because, first it measures the association between two random variables, second, unlike the linear correlation, it takes in account all the possible kinds of correlation between the random variables. The weak point of the mutual information is the estimation: if the number of data is small, it is difficult to estimate precisely the information theory functions. This fact is well developed in [8] or [9]. The mutual information is a real value that represents how much information on Y we can get if we know X (and vice-versa), or how much information is shared by X and Y.

$$I(X;Y) = H(X,Y) - H(X \mid Y) - H(Y \mid X), \tag{4}$$
$$I(X;Y) = H(X) + H(Y) - H(X,Y). \tag{5}$$

From that, we can see that the mutual information can be the base of the creation of a good functions that measure the degree of association between two variables. For more information about these previous information theory definitions, please refer to [10] or [11].

3 Distances Based on Information Theory

One of the most important features in the design of an unsupervised classification algorithm is the choice of the notion of distance. This choice defines the notion of closeness for the studied elements, it is important to understand that different choices for this notion can lead to completely different classifications. This section introduces information theory based functions that measure the inverse of the correlation: Subsection 3.1 deals with normal distances and Subsection 3.2 gives normalized distances. Although nearly any function can be chosen to define a notion of distance, usually it is better to use a distance function that has a precise definition in Mathematics:

Definition 2. *$d : E \times E \to \Re$ is a distance over E if:*

- *d is positive: $d(x, y) \geq 0$ with equality if and only if $x = y$*
- *d is symmetric: $d(x, y) = d(y, x)$*
- *d verifies the triangular inequality: $d(x, y) \leq d(x, z) + d(z, y)$*

3.1 Distances

In this part, we will define two information theory based distances over the random variables.

$$D_1(X,Y) = H(X,Y) - I(X;Y) = H(X \mid Y) + H(Y \mid X), \tag{6}$$
$$D_2(X,Y) = max(H(X), H(Y)) - I(X;Y) = max(H(X \mid Y), H(Y \mid X)). \tag{7}$$

It is easy to see that these two functions measure the inverse of the correlation between X and Y. If we take two random variables X and Y, the more information they share, the less the distance between them is. When these two variables are equal: complete correlation, the distance is zero. The measurements D_1 and D_2 are mentioned in [10] and [12], the following paragraphs show that they are distances.

Property 1. D_1 and D_2 both are distances over the random variables.

Proof. For both D_1 and D_2 the positivity and the symmetry are quite obvious. For the triangular inequality, with D_1:

$$\begin{aligned} D_1(X,Y) &= H(X \mid Y) + H(Y \mid X) \leq H(X,Z \mid Y) + H(Y,Z \mid X) \\ &= H(Z \mid Y) + H(X \mid Y,Z) + H(Z \mid X) + H(X \mid Y,Z) \\ &\leq H(Z \mid Y) + H(X \mid Z) + H(Z \mid X) + H(X \mid Z) \\ &= D_1(X,Z) + D_1(Z,Y). \end{aligned}$$

So D_1 verifies the triangular inequality. For D_2, we need to proceed with a disjunction of cases on the values of the entropy of X, Y, Z.

- If $H(X) \geq H(Y) \geq H(Z)$:

$$\begin{aligned} D_2(X,Y) &= H(X \mid Y) = I(X;Z \mid Y) + H(X \mid Y,Z) \\ &\leq H(Z \mid Y) + H(X \mid Z) \leq H(Y \mid Z) + H(X \mid Z) \\ &= D_2(X,Z) + D_2(Z,Y). \end{aligned}$$

- If $H(X) \geq H(Z) \geq H(Y)$:

$$\begin{aligned} D_2(X,Y) &= H(X \mid Y) = I(X;Z \mid Y) + H(X \mid Y,Z) \\ &\leq H(Z \mid Y) + H(X \mid Z) = D_2(X,Z) + D_2(Z,Y). \end{aligned}$$

- If $H(Z) \geq H(X) \geq H(Y)$:

$$\begin{aligned} D_2(X,Y) &= H(X \mid Y) \leq H(Z \mid Y) \\ &\leq H(Z \mid Y) + H(Z \mid X) = D_2(X,Z) + D_2(Z,Y). \end{aligned}$$

- The cases where $H(Y) > H(X)$ are symmetric.

So D_2 verifies the triangular inequality. □

3.2 Normalized Distances

To compare different distances or to compare the distance between two unions of variables that do not have the same size, it could seem normal to use normalized distances. This is the normalized forms of D_1 and D_2

$$d_1(X,Y) = 1 - \frac{I(X;Y)}{H(X,Y)}, \tag{8}$$

$$d_2(X,Y) = 1 - \frac{I(X;Y)}{max(H(X),H(Y))}. \tag{9}$$

Property 2. d_1 and d_2 both are normalized distances over the random variables.

Proof. The paper [6] provides a good demonstration. □

So we have seen four possible information theory based distances over random variables (for more properties about information measure, refer to [12]). And as an union of random variables is still a random variable, we can also compute the distances between unions of random variables. But, in practice for clustering, it is better to use distances over clusters: we will see that even if clusters and unions of random variables both deal with groups of random variables, still they are a bit different.

4 Classification Using Information Distances

This part explains how unsupervised classification can be achieved using the information theory distances of the previous section. The random variables of the starting set have to be regrouped using an information distance. The strongly correlated variables should be classified in the same clusters, and the independent ones should not be together. Subsection 4.1 explains the extension of the information distances to clusters. In Subsection 4.2, we develop some classification methods.

4.1 Equality and Distances for Clusters

The information theory based distances that we have seen so far are distances over random variables or random vectors. But in fact, for classification, we need distances over clusters. **To understand the difference between clusters and unions of random variables is important**. Although both express the idea of a group of random variables: set of random variables and union of random variables, the difference still exists. This difference comes from their different definition of equality. Two clusters are equal if and only if they are constituted with exactly the same elements. Two unions of random variables are equal if and only if there is a bijection between them.

Definition 3. *In classification of the variables $X_1, \ldots, X_n$, we say that two clusters are equal if they are composed of the same variables.* $\forall G_1 \times G_2 \subseteq [1, n]^2$*:*

$$\{X_i \mid i \in G_1\} := \{X_i \mid i \in G_2\} \Leftrightarrow G_1 = G_2. \tag{10}$$

For example, if X, Y, Z are three random variables such that $X \sim Y$ then the two unions of random variables satisfy: $(X, Z) \sim (Y, Z)$. But the clusters $\{X, Z\}$ and $\{Y, Z\}$ are not the same. Then, with all the information distances that have been studied in the previous section, the distance between these two clusters $\{X, Z\}$ and $\{Y, Z\}$ is zero, because the random variables are equal, but these clusters are different: impossible. Thus, the information distances of the previous section are not distances over the clusters. But if the random variables satisfy a simple hypothesis, we can continue to use these information distances with clusters.

Lemma 1. *For the random variables:* $X_1, \ldots, X_n$ *such that*

$$\forall k \in [1,n], H(X_k \mid X_{[1,n]\setminus\{k\}}) \neq 0, \tag{11}$$

we have

$$\forall G_1 \times G_2 \subseteq [1,n]^2, \{X_i \mid i \in G_1\} = \{X_i \mid i \in G_2\} \Leftrightarrow X_{G_1} \sim X_{G_2}. \tag{12}$$

That is to say: clusters' equality $\Leftrightarrow$ *random variables' equality. Secondly, the reciprocal is also true: the equivalence (12) implies the condition (11).*

Proof. For the first part of the lemma, let's suppose that $\forall k \in [1,n], H(X_k \mid X_{[1,n]\setminus\{k\}}) \neq 0$. As we know that in any case:

$$< 1 > \text{ clusters' equality} \Rightarrow \text{random variables' equality.}$$

And if $G_1 \neq G_2$ such that G_1 is not included in G_2, let $k \in G_1 \setminus G_2$ then $H(X_k \mid X_{[1,n]\setminus\{k\}}) \neq 0$ and $H(X_{G_1} \mid X_{G_2}) > 0$. So X_{G_1} is not equal to X_{G_2}. We have just showed:

$$< 2 > \text{ no clusters' equality} \Rightarrow \text{no random variables' equality.}$$

With $< 1 >$ and $< 2 >$ we have the equivalence in (12). For the second part of the lemma, we assume that: $\exists k \in [1,n], H(X_k \mid X_{[1,n]\setminus\{k\}}) = 0$. Let $G_1 = [1,n] \setminus \{k\}$ and $G_2 = [1,n]$ then $X_{G_1} \sim X_{G_2}$ but $\{X_i \mid i \in G_1\} \neq \{X_i \mid i \in G_2\}$. So, in this case, the equivalence in (12) is not true. Thus (12) implies (11). □

Therefore, a necessary and sufficient condition for the equivalence of the clusters' equality and the random variables' equality is no variable is a function of the other variables. The interest is that we can have distances over clusters under the condition (11).

Theorem 1. *Under the hypothesis (11) of Lemma 1,* D_1, D_2, d_1, d_2 *are distances over the clusters of* $\{X_1, \ldots, X_n\}$

Proof. In fact, with clusters, everything goes in the same way as with unions of variables, the only problem was the first property of the distances. Without the supposition of Lemma 1, we can have two different clusters with a distance between them equal to zero. But if we use the hypothesis of Lemma 1, we know that this can not happen anymore. So D_1, D_2, d_1, d_2 are distances over clusters. □

That means that each variable has to add a bit of randomness or information to the system. This theorem suggests that, before classification, the variables that are function of the others have to be removed from the starting set. Then, we can do the clustering. The classification is done on the causes of the system, it is no use to classify the other variables, because they do not add any meaning to the system. And if we do the classification with these variables, it is like taking twice in account the same information: it risks to unbalance the classification.

4.2 Hierarchical Classification Algorithm

Once a notion of distance is fixed, there still a choice to make: the algorithm that will use the distance to separate data into clusters. The hierarchical classification algorithm works in this way:

- Each element of the starting set begins in its own cluster.
- Construct the distance matrix of the clusters (all the cluster to cluster distances).
- Find two clusters such that their distance is the minimum of the distance matrix (if the minimum is achieved by more than one choices, just select one).
- Merge these two clusters into a new cluster.
- Update the distance matrix.
- While the number of clusters is greater than 1, go back to the third step.

The outcome of this algorithm is a binary tree: its leafs are all the elements of the starting set, and each other node corresponds to the merging of two clusters. Then, any horizontal cut of this tree gives a classification of the variables. For more details about Hierarchical Classification, refer to [5] or [13]. This is an empirical algorithm, so in some cases, this algorithm will not produce the best solution, it will just produce a good solution.

5 Application

Subsection 5.1 explains the studied system. Then, in Subsection 5.2, we compare two hierarchical algorithms that use the same distance, the difference is that one accepts the hypothesis of Lemma 1 and the other does not.

5.1 The Data

This part shows how the classification works in practice. For convenient reasons, we use simulated data. This is the study of a quite complex system that contains ten binary random variables, the system is known. The idea is not really to study this system but it is to compare the classification methods applied on it. Fig. 1 shows how this system is constructed and what the relations between the different variables are. The system has some entries, they are all binary, independent and uniformly distributed random variables (thick arrows on the graph), they are generated by Matlab. For example, the variables X_1 and X_2 are equal to two of these entries. Then, combinations of these entries give the other variables of the system: $X_3 = X_1 \; xor \; X_2$. The variables X_4, X_5 and X_6 are constructed together, they are all the combination (*and* operation) of a multiple entry (the triple arrow in the center) and another entry. We also see that the variable X_7 is a function of X_3, X_4, X_5, X_6, this function is defined by this equation: $X_7 = X_3 \wedge (X_4 \vee (X_5 \; xor \; X_6))$. In the end, we can notice that the last variables X_7, X_8, X_9, X_{10} form a Markov Chain ($X_7 \rightarrow X_8 \rightarrow X_9 \rightarrow X_{10}$).

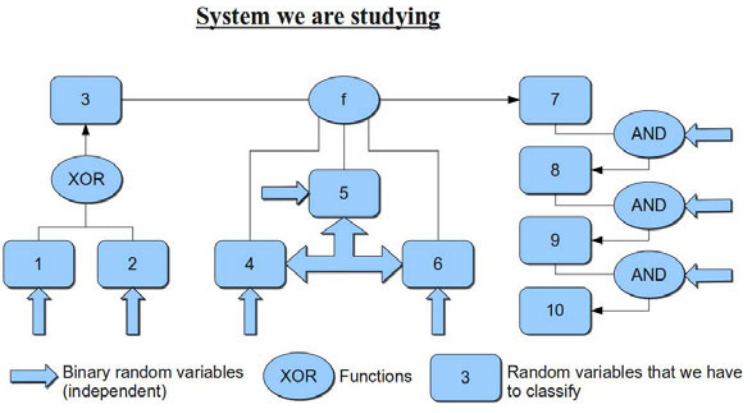


Fig. 1. Diagram of the studied system

At first sight, this diagram shows that the system seems to be divisible in three part:

$$\{X_1, X_2, X_3\}, \{X_4, X_5, X_6\}, \{X_7, X_8, X_9, X_{10}\}. \tag{13}$$

We can hope that the classifications will show this repartition.

5.2 Hierarchical Classification

This part compares the results of two different hierarchical classifications, both these classifications use the normalized distance d_1, the difference relies on the acceptation or non-acceptation of the hypothesis of Lemma 1. This hypothesis is that no variable should be a function of the other variables.

The first classification does not satisfy this hypothesis, it takes the ten variables, builds the distance matrix using d_1 and proceeds to classification following the hierarchical algorithm that was presented in Section 4.2. This is the method that was given in [5] and [6]. The binary tree of the classification is given in Fig. 2.

This classification is not bad, the variables X_1 and X_2 are both pairwise independent with the other variables, they are merged the last with the other

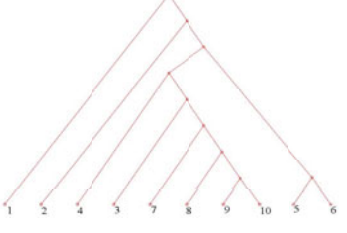


Fig. 2. The first hierarchical classification with all the variables

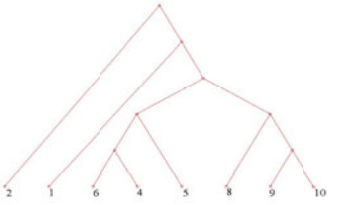


Fig. 3. The second hierarchical classification with hypothesis of Lemma 1

variables. And if we just look at the Markov Chain: variables X_7 to X_{10}, the classification follows the order of the chain. The problem is that the division of the system in subsystems is not clear (compare with (13)).

The second classification takes in account the hypothesis of Lemma 1. We quickly find out that X_7 is a function of X_3, X_4, X_5, X_6 and so has to be removed, so does X_3 which is a function of X_1, X_2. Another classification is made using the same distance d_1 but only with the remaining variables. This time, we obtain Fig. 3.

This classification has the same properties as the previous one, but it has a good advantage: at the first sight, this tree shows the separation of the system in three parts. As X_3 is a function of X_1, X_2, we put them together, then the tree shows that X_4, X_5, X_6 work together, so do X_8, X_9, X_{10}. The role of X_7 can then be seen as a kind of link between these three groups: X_7 is a function of the first two groups, and but it is also correlated to the last group. So from the point of view of the meaning (see (13)), the classification that accepts the hypothesis of Lemma 1 is better than the other. It is also better in computation time.

6 Conclusion

To conclude, four information distances over the random variables have been provided. Then, this paper proves, in Theorem 1, that it is possible to extend this distances to clusters with the hypothesis of Lemma 1. We also have explained some classification methods. Finally, some practical applications has been run on Matlab, they explain and compare different classification methods. And they show, in practice, the advantage of the hypothesis of Lemma 1.

But there is still a lot to do. First, we can improve the algorithm: find other classification algorithms and compare them to the hierarchical classification. Second, if we want to use the information theory based classification methods with real data, we need to improve the functions that estimate the information theory distances.

Acknowledgements. This work was supported by the National Natural Science Foundation of China under Grants 60972033 and 60832001, and the National Basic Research Program of China under Grant 2007CB310900.

References

1. Papoulis, A., Pillai, S.U.: Probability, Random Variables, and Stochastic Processes, 4th edn. McGraw Hill, New York (2002)
2. Bansal, N., Blum, A., Chawla, S.: Correlation Clustering. Machine Learning 56, 89–113 (2004)
3. Glynn, E.F.: Correlation Distances and Hierarchical Clustering. TechNote, Stowers Institute for Medical Research (2005)
4. Tishby, N., Pereira, F., Bialek, W.: The Information Bottleneck Method. In: 37th Annual Allerton Conference on Communication, Control and Computing, pp. 368–377. IEEE Press, Monticelo (1999)
5. Kraskov, A., Grassberger, P.: MIC: Mutual Information Based Hierarchical Clustering. In: Emmert-Streib, F., Dehmer, M. (eds.) Information Theory and Statistical Learning, pp. 101–124. Springer, New York (2009)
6. Kraskov, A., Grassberger, P.: Hierarchical Clustering Based on Mutual Information. Europhysics Letters 70(2), 278–284 (2005)
7. Shannon, C.E.: A Mathematical Theory of Communication. Bell System Technical Journal 27, 623–656 (1948)
8. Kraskov, A., Stogbauer, H., Grassberger, P.: Estimating Mutual Information. Physical Review E 69, 066138 (2004)
9. Paninski, L.: Estimation of Entropy and Mutual Information. Neural Computation 15, 1191–1253 (2003)
10. Cover, T.M., Thomas, J.A.: Elements of Information Theory, 2nd edn. Wiley, New Jersey (2006)
11. Yeung, R.W.: Information Theory and Network Coding. Springer, New York (2008)
12. Yao, Y. Y.: Information-Theoretic Measures for Knowledge Discovery and Data Mining. In: Karmeshu (eds.) Entropy Measures, Maximum Entropy Principle and Emerging Applications, pp. 115–136. Springer, Heidelberg (2003)
13. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall, New Jersey (1988)

Regression Transfer Learning Based on Principal Curve

Wentao Mao, Guirong Yan, Junqing Bai, and Hao Li

The Key Laboratory of Strength and Vibration of Ministry of Education
Xi'an Jiaotong University, Xi'an, China
maowt.mail@gmail.com

Abstract. One of the basic ideas of present transfer learning algorithms is to find a common underlying representation among multiple related tasks as a bridge of transferring knowledge. Different with most transfer learning algorithms which are designed to solve classification problems, a new algorithm is proposed in this paper to solve multiple regression tasks. First, based on "self-consistency" of principal curves, this algorithm utilizes non-parametric approach to find the principal curve passing through data sets of all tasks. We treat this curve as common-across-tasks representation. Second, the importance of every sample in target task is determined by computing the deviation from the principal curve and finally the weighted support vector regression is used to obtain a regression model. We simulate multiple related regression tasks using noisy Sinc data sets with various intensities and report experiments which demonstrate that the proposed algorithm can draw the useful information of multiple tasks and dramatically improve the performance relative to learning target task independently. Furthermore, we replace principal curve with support vector regression with model selection to find common representation and show the comparative results of these two algorithms.

Keywords: Transfer learning, Support vector machine, Principal curve.

1 Introduction

Traditional machine learning algorithms are constructed under the assumption that the training and test data have same distribution and feature sets. But in practical engineering, this fundamental assumption is often violated. In recent years, transfer learning has been studied to solve this problem. The goal of transfer learning is to improve learning performance in target domain by reusing knowledge learnt in some related source domain, which has demonstrated impressive performance in the fields of text recognition[1] and sensor network[2], etc.

According to [3], there are two main approaches to transfer learning in the past. One is referred to as *instance-transfer* ,in which training instances in source domain are re-weighted in the light of the similarity to the target domain. For example, Dai[4] aimed at solving the learning problem with a small number of target-domain data and a lot of source-domain data. The author utilized AdaBoost as basic learning tool to find the useful data in source-domain by iteratively adjusting their weights. Shi[5]firstly constructed a classifier trained only in target domain to measure the similarity of source instance to target instances and then set higher weights on similar instances in k-NN. The other is *feature-representation-transfer* , in which the common methods are to find

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 365–372, 2010.

a low-dimensional representation across multiple domains. As a representational work, Argyrious[6] adopted 1-norm regularization to control the number of common features and transformed it to a convex optimization using trace-norm regularization. Finally the common-across-tasks representation was found. Pan[7] learned a low-dimensional latent feature space by which the useful knowledge of source domain can be transferred to target domain. The key part of this algorithm is to find the same or close distributions between source and target domains. It is worth noting that most of present transfer learning algorithms aim at solving classification problems, while only few of literatures[6,7] considered regression problem. Inspired by [6,7,5], we combine the characteristics of these two kinds of approaches and present a simple but efficient transfer learning algorithm. It has a plain interpretation: it performs a non-parametric analysis and a supervised step in order, where in the former step we learn common-across-tasks representations via principal curve, which keeps in step with *feature-representation-transfer*, and in the latter step we learn task-specific functions using weighted support vector regression, which approximates *instance-transfer*. We choose non-parametric rather than parametric analysis to seek internal structure of multiple related tasks because there often is little priori knowledge about data distribution and causative connection between tasks. As a rule of thumb, we should explore the best way to describe the regular pattern of data directly from data itself, without any pre-defined parameters. This paper is organized as follows. In Section 2, we provide a brief review to principal curve. In Section 3, the idea and design of transfer learning algorithm based on principal curve are elaborated in detail. Experimental results on simulated Sinc data sets are then presented in Section 4, followed by a conclusion of the paper in last section.

2 Theory of Principal Curve and K Curve

In 1983, Hastie[8] firstly introduced the theory of principal curve. Afterwards, this theory were successfully applied to solve practical problems, like data visualisation[9] and ecology analysis[10], etc. Principal curve is the extension of principal component analysis and its basic idea is to find a continuous one-dimensional manifold that approximate the data in the sense of "self-consistency", i.e. the curve should coincide at each position with the expected value of the data projecting to that position[8]. Intuitively, this curve passes through the "middle" of a high-dimensional data set. The difference between principal curve and regression is that a non-parametric method is used to explore the trajectory in data set, without any assumption about causal relationships among instances[11]. Hastie argued that principal curve can truly reflect the shape of data set, i.e. the curve is skeleton and the data set is cloud[8].

In this paper, we choose K curve for its good practicability. Kégl[12] proved that for any data set with finite second moments there always exists a principal curve. The definition of K curve is listed as follows.

Definition 1 **(K principal curve[13]).** *For a data set $X = \{x_1, x_2, \cdots, x_n\} \subset \mathbb{R}^d$, a curve f^* is called a K principal curve of length L for X if f^* minimizes $\triangle(f)$ over all curves of length less than or equal to L, where f is a continuous function $f : I \to \mathbb{R}^d$, $\triangle(f)$ is the expected squared distance between X and f and defined as:*

$\triangle(f) = E[\triangle(X, f)] = E[\inf_\lambda \|X - f(\lambda)\|^2] = E[\|X - f(\lambda_f(X))\|^2]$
where $\lambda_f(x) = \sup\{\lambda : \|X - f(\lambda)\| = \inf_\tau \|x - f(\tau)\|\}$ *is called projection index.*

The goal of K curve is to find a set of polygonal lines with K segments and with a given length to approximate the principal curve. The algorithm is based on a common model about complexity in statistical learning theory[12]. The framework of this algorithm can be summarized as follows. At the beginning, $f_{1,n}$ is initialized by the first principal component line and in each iteration step, a new vertex is added on $f_{i-1,n}$ which is obtained in $i-1$ step, to increase the number of segments. According to the principle of minimizing the projection distance, the position of vertexes are optimized to construct a new curve $f_{i,n}$. Kégl[12] gave a detailed description of this algorithm.

3 Transfer Learning Based on Principal Curve

If we treat the data of multiple related tasks as features of a data set, then as suggested by [6], the important point becomes how to find the bridge of transferring knowledge, i.e. the common low-dimensional representations. For regression problems, this problem equals to find the low-dimensional skeleton of data clouds of multiple tasks, which is in complete agreement with "self-consistency" of principal curve in the sense of 1-dimensional manifold. Moreover, we usually lack enough priori knowledge about the internal pattern of tasks' data. So the better choice is choosing non-parametric method to analyze multiple regression tasks. However, there are differences between the method of using principal curve to find across-tasks representations and the approaches proposed in [6,7]. Comparatively speaking, the former focuses on depicting the shape of data set, for example, principal curve is in 1-dimensional space and principal surface is in low-dimensional space which are not suited for direct modeling, while the latter is inclined to reconstruct target data by means of the latent low-dimensional space, and then model the target task using present learning algorithms. Therefore, it needs to consider how to transfer the knowledge represented by principal curve. The strategy we choose is assigning different weights on samples in learning algorithm in the light of the deviation from principal curve, which is inspired by [13]. The values of weights are relevant to the degree of deviation. We adopt weighted solution path algorithm of support vector regression proposed in [14] as learning algorithm.

We firstly analyze the rationality of using principal curve to represent the common part of multiple tasks. Qi[13] gave the following theorem:

Theorem 1. *Let* $D \subseteq \mathbb{R}^d$ *be a sample space and* W *is a set of parameters. Let* $T \subseteq D$, $X = \{x_1, x_2, \cdots, x_d\} \in T$, $w \in W$. $\lambda_f(X)$ *is defined in Definition 1. If* $f(\lambda)$ *is a principal curve and let* $w = \lambda_f(X)$, *it holds that:*
$\sum_{j=1}^d Var(x_j) = E\|X - f(\lambda_f(X))\|^2 + \sum_{j=1}^d Var(f_j(\lambda_f(X))$

The proving procedure of Theorem 1 was given in detail in [13].

In the case of multiple related tasks, T is the assembly of all tasks' data. Then Theorem 1 means the data of every task is sum of two parts where the first one is estimate variance caused by principal curve which passes through the middle of T, and the other is the residual variance in the sense of the expected squared distance from a sample in

target task to its projection position on the curve. It indicates principal curve can represent the across-tasks knowledge of multiple related tasks.

The details of transfer learning algorithm based on principal curve are described as follows.

Algorithm1:
Input: source data sets $S = \{(x_{ti}, y_{ti})\}_{i=1}^{m}$,$t \in \{1, \cdots, T\}$($T$ is the number of source tasks), target training set $\{(x_i, y_i)\}_{i=1}^{p}$ and test set $\{(x_i, y_i)\}_{i=1}^{q}$
Parameters: regularization parameter λ, Gaussian kernel width σ
Output: regression output $Y = (y_1, \cdots, y_m)$
Initialization: set f is the first principal component line of S
Use the algorithm in [12] to compute f as the K-principal curve of S
Use a cubic spline to interpolate f and obtain a new curve f^*
for $i = 1, \cdots, p$
compute $distance_i = \frac{\|x_i - f^*(\lambda_f(x_i))\|}{\|x_i\|}$
end for
Arrange all samples in descending sequence of $distance_i$,$i = 1, 2, \cdots, p$
Utilize the weighted solution path algorithm of support vector regression[14] to model the training set. Exponential weight function which is introduced in [15] is chosen to produce the weights and described as follows:
$w(i) = \frac{1}{1+exp(a-2ai/l)}$ where a is the parameter to control the ascending rate.
Finally predict test set.
End Algorithm 1.

4 Experimental Results

In order to test the effectiveness of Algorithm 1, we choose noisy Sinc data set as regression task to model. First, the data sets $\{(x_i, y_i)\}$ of 10 tasks are generated with x_i drawn uniformly from $[-3, +3]$ and $y_i = sin(x_i)/(x_i) + e_i$, where e_i is a Gaussian noise term with zero mean and variance from 0.1 to 1.0 with interval of 0.1. We assign the number of these tasks from 1 to 10 in ascending sequence of variance. Each task consists of 100 samples. In each task, 70 samples are randomly selected as training set and others are for testing. Obviously these 10 tasks are highly related. The Gaussian RBF kernel is thus used and defined as $K(x_i, x_j) = exp(\|x_i - x_j\|^2/\sigma)$.

In this section Algorithm 1, called here TLPCURVE, is tested in comparison with two methods: one is learning each task independently using $\epsilon - Path$[16], called here NoTL, and another is replacing principal curve in Algorithm 1 with support vector regression(SVR), called here TLSVM. The reason of comparing TLSVM is to test the performance of parametric analysis. In order to get the best performance of TLSVM, we adopt the model selection method in [17] to find the regression curve with best generalization. For TLPCURVE and NoTL, the regularization parameter λ is set 0.01 and Gaussian kernel width σ is set 0.02. $RMSE = \sqrt{\frac{1}{q}\sum_{i=1}^{q}|\hat{y_i} - y_i|}$ on the test set is computed, where $\hat{y_i}$ is the value of prediction and $y_i = sin(x_i)/x_i$ with no noise. A method with lower RMSE is better.

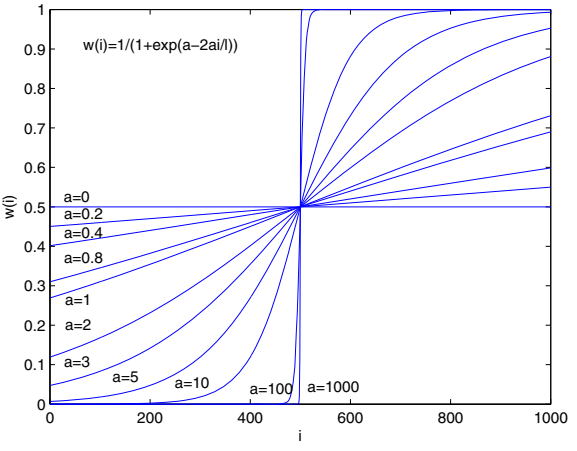


Fig. 1. Exponential weight function with different values of a

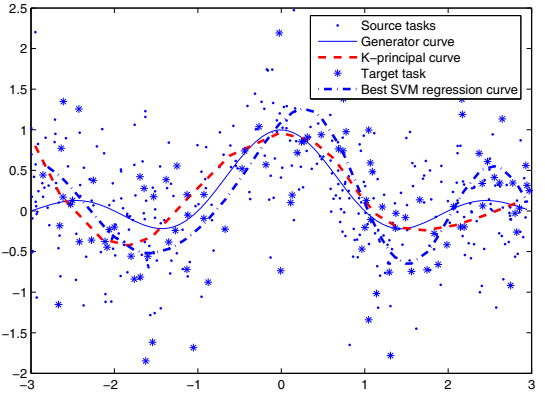


Fig. 2. The modeling performance of TLPCURVE and TLSVM

As pointed out by [15], different values of a affect the performance of forecasting intensively. In our experiments, the parameter a of exponential weight function is set 0.2, 1, 10 to test this functions performance. Just like [15], we depict the weight curves with various a in Fig.1 for more clear illustration. Moreover, linear weight function is also introduced for comparison and described as[15]: $w(i) = \frac{i}{l(l+1)/2}$.

First, we choose the former 6 tasks to test, where task 5 is set as target task and others are source tasks. a is set 10. Fig.2 shows the modeling performances of TLPCURVE and TLSVM.

As priori knowledge, Sinc curve is common part among these 6 tasks. As illustrated in Figure 1, although the randomicity of noise negatively affect the performance of two methods, principal curve (dashed line) deviates Sinc curve(solid line) less than the best regression curve obtained after model selection. It confirms again that in the case of

Table 1. Mean error and standard deviation of 30 RMSEs

	TLPCURVE	TLSVM	NoTL
Exponential(a=10)	0.32226(0.05048)	0.37621(0.07245)	0.39307(0.10383)
Exponential(a=1)	0.40193(0.08547)	0.39953(0.08663)	0.39307(0.10383)
Exponential(a=0.2)	0.39537(0.08421)	0.42269(0.13277)	0.39307(0.10383)
Linear	0.38305(0.11550)	0.37340(0.08726)	0.39307(0.10383)

lacking enough priori knowledge about data sets, non-parametric methods which explore the internal structure directly from data perhaps perform better than parametric methods. The comparative results also suggest the effectiveness of TLPCURVE whose RMSE is 0.2675 versus 0.5481 of NoTL and 0.3283 of TLSVM.

For the sake of reducing the negative effect of randomicity of noise, we repeat generating all ten tasks' data and implement three methods 30 times. We then report the average RMSE and standard deviation for the 30 RMSEs on predicting test data in Table 1.

As listed in Table 1, the mean error of TLPCURVE with a=10 is lower than NoTL and TLSVM as well as standard deviation value(in bracket). Moreover, the performance of TLSVM is better than NoTL. In addition, the errors of TLPCURVE with a=0.2 and linear weight-setting are both lower than NoTL. As discussed above, the key question of obtaining good performance is to choose a proper parameter of weight function. Faulty weights will deteriorate performance, just as the exponential weights with a=1 and a=0.2 in Table 1. It is obvious that a=10 is much better than other three weight-setting methods. As shown in Fig.1, the curves with a=1 and a=0.2 are close to straight line, which cannot reflect the variation tendency of samples' importance well. As a result, a is always set 10 in the rest of this paper. The comparative results indicate that principal curve is fit for acting as a bridge of transferring knowledge and can improve the learning performance. Meanwhile, TLSVM cannot exploit the knowledge of data entirely and then has to get an inferior performance than TLPCURVE.

In order to test the effectiveness of Algorithm 1 on different tasks, we produce an individual data set and select a task as target task and set other 9 tasks as source tasks. The parameters are same as above. Fig.3 demonstrates the comparative results of three methods.

As shown in Fig.3, TLPCURVE gets much lower generalization error than NoTL on almost all tasks, except for task 1 and 9. This is because learning independently on task 1 with smallest noise can obtain a regression curve which is very close to sinc curve. And the randomicity of noise in former tasks interrupts the transfer learning for task 9. In addition, the performance of TLPCURVE is apparently better than TLSVM only on task 2,3,5,7,8. It indicates that larger deviation of noise in target task will lead to invalid regression model of SVR. It is worth noting that the generalization error of TLSVM is higher than NoTL on task 2,3,5 which indicates that unreasonable mining the internal structure of training data sets could inversely cause bad performance, and demonstrates again that principal curve is a efficient way to find the across-tasks representation.

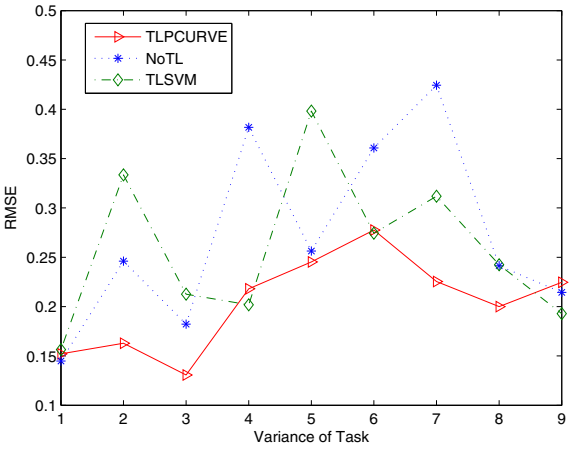


Fig. 3. Comparative results of three methods on different target task

5 Conclusion

In this paper, we have presented a new transfer learning algorithm which utilizes principal curve to seek the common-across-tasks representation among multiple regression tasks. We choose weighted support vector regression to transfer common knowledge from principal curve to target task through measuring deviation of target data from the curve. The whole analysis of this paper is based on the characteristic "self-consistency" of principal curve which make the common part of multiple regression tasks be truly represented using non-parametric methods. It is worth noting that principal curve can depict data set with special structure (for example, ring shape) much better than SVR and other supervised methods, which is the primary reason we choose it. This work may be extended in different directions. First, the experimental results in Table 2 demonstrate that the weight-setting method is crucial to improve the learning performance of the proposed algorithm. So we need to generate a group of more effective weights by analyzing the correlation among various tasks. Moreover, the simulation example in Section 4 is in two-dimensional space, and the proposed algorithm is also applicable to multi-dimensional data sets. So one solution is replacing principal curve with principal surface whose dimension is greater than 1. Because the unit of area on surface doesn't act globally, how to build a reasonable theory about local equivalence will be a difficult point.

Acknowledgement

We thank the author Zhang of literature[11] for providing source code of K-principal curve.

References

1. Dai, W., Xue, G., Yang, Q., Yu, Y.: Co-clustering based classification for out-of-domain documents. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 210–219. ACM, New York (2007)
2. Yin, J., Yang, Q., Ni, L.M.: Adaptive temporal radio maps for indoor location estimation. In: Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications (PerCom 2005), pp. 85–94 (2005)
3. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering 99 (2009)
4. Dai, W., Yang, Q., Xue, G., Yu, Y.: Boosting for transfer learning. In: Proceedings of the 24th International Conference on Machine Learning (ICML), pp. 193–200. ACM, New York (2007)
5. Shi, Y., Wang, J.: Similarity-based weighting approach for transfer learning. Technical report, Sun Yat-sen University, China (2009)
6. Argyriou, A., Evgeniou, T., Pontil, M.: Multi-task feature learning. In: Proceedings of the 19th Annual Conference on Neural Information Processing Systems (NIPS), pp. 41–48. MIT Press, Cambridge (2007)
7. Pan, S.J., Kwok, J.T., Yang, Q.: Transfer learning via dimensionality reduction. In: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, pp. 677–682. AAAI Press, Menlo Park (2008)
8. Hastie, T.: Principal curves and surfaces. Technical report, Department of Statistics, Stanford University (1983)
9. Hermann, T., Meinicke, P., Ritter, H.: Principal curve sonification. In: Proceedings of International Conference on Auditory Display, International Community for Auditory Displayn, pp. 81–86 (2000)
10. De'ath, G.: Principal curves: A new technique for indirect and direct gradient analysis. Ecology 80, 2237–2253 (1999)
11. Zhang, J., Wang, J.: An overview of principal curves. Chinese Journal of Computers 26, 129–146 (2003)
12. Kégl, B., Krzyzak, A., Linder, T., Zeger, K.: Learning and design of principal curves. IEEE Transactions on Pattern Recognition and Machine Intelligence 22, 281–297 (2000)
13. Qi, H., Wang, J.: A model for mining outliers from complex data sets. In: Proceedings of the 19th Annual ACM Symposium on Applied Computing, pp. 595–599. ACM, New York (2004)
14. Mao, W., Dong, L., Zhang, G.: Weighted solution path algorithm of support vector regression for abnormal data. In: Proceedings of the 19th International Conference on Pattern Recognition (ICPR 2008), pp. 1–4 (2008)
15. Tay, F., Cao, L.: Modified support vector machines in financial time series forecasting. Neurocomputing 48, 847–862 (2003)
16. Wang, G., Yeung, D.Y., Lochovsky, F.H.: A new solution path algorithm in support vector regression. IEEE Transactions on Neural Networks 19(10), 1753–1767 (2008)
17. Hu, D., Mao, W., Zhao, J., Yan, G.: Application of lssvm-pso to load identification in frequency domain. In: Proceedings of Artificial Intelligence and Computational Intelligence, pp. 231–240. Springer, Heidelberg (2009)

Semivariance Criteria for Quantifying the Choice among Uncertain Outcomes

Yankui Liu* and Xiaoqing Wang

College of Mathematics & Computer Science, Hebei University
Baoding 071002, Hebei, China
yliu@hbu.edu.cn, wxqwyzy@163.com

Abstract. In a stochastic decision system, mean-risk is an approach frequently used for modeling the choice among random outcomes, the method quantifies a risk management problem by two criteria (i.e., mean and risk) with possible trade-off analysis. In the literature, there are different risk definitions for a random variable such as variance, critical probability and stochastic dominance. This paper presents semivariance of fuzzy random variable as a new risk criteria for measuring hybrid uncertain outcomes. Since the semivariance is defined by nonlinear fuzzy integral, its computation is a challenge issue for research, and usually depends on intelligent algorithms. This paper will develop some useful semivariance formulas for common triangular and trapezoidal fuzzy random variables, which have potential applications in various practical risk management problems.

Keywords: Mean, Risk, Semivariance, Fuzzy random variable.

1 Introduction

The seminal Markowitz portfolio optimization model [1] employs the variance as the risk measure in mean-risk analysis. Since then, the method has been used and extended by a number of researchers in the literature. Yan *et al.* [2] used semivariance in Markowitz portfolio model and extended one period portfolio to multi-period; Ogryczak and Ruszczyński [3] showed that the standard semideviation as the risk measure makes the mean-risk model consistent with the second degree stochastic dominance; Wagner *et al.* [4] considered a location-optimization problem, in which the classical uncapacitated facility location model is recast in a stochastic environment with several risk factors; Chang *et al.* [5] introduced a heuristic approach to portfolio optimization problems in different risk measures and compared its performance to mean-variance model, and Graves and Ringuest [6] gave a tutorial that demonstrated the current state-of-the-art methods for incorporating risk into project selection decision making.

As an extension of random variable, fuzzy random variable was first proposed by Kwakernaak [7], it has been extended by a number of researchers

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 373–380, 2010.

in fuzzy community, and became a popular tool to deal with twofold uncertainty in a decision system. For recent development about the study of fuzzy random variable as well as its applications, the interested reader may refer to [8,9,10,11,12,13,14,15,16,17]. The purpose of this paper is to present semivariance of a fuzzy random variable as a new risk criteria for measuring hybrid uncertain outcomes. Our motivation for this work is as follows. In the literature, the use of the semivariance of random variable rather than variance as the risk measure was suggested by Markowitz [18], and it was shown that the mean-risk model using a fixed-target semivariance as the risk measure is consistent with stochastic dominance. In addition, just as pointed out by Estrada [19], the variance of random returns is an appropriate measure of risk only when the underlying probability distribution of returns is symmetric, and it can be applied straightforwardly as a risk measure only when the underlying probability distribution of returns is Normal. On other hand, the semivariance of random returns is a more plausible measure of risk because the semivariance is more useful than the variance when the underlying probability distribution is asymmetric and just as useful when the underlying probability distribution is symmetric. More important, investors obviously do not dislike upside volatility, they only dislike downside volatility. However, compared with the stochastic situation, the computation for the semivariance of a general fuzzy random variable is difficult due to its definition including nonlinear fuzzy integral. Hence, in this paper, we consider the case when the fuzzy random variables are triangular and trapezoidal, for which the exact computational formulas can be derived.

The plan of the paper is as follows. Section 2 defines the semivariance of a fuzzy random variable and discusses its basic properties. In Section 3, for any given random outcome, we compute the quadratic semi-deviation for triangular and trapezoidal fuzzy variables. Based on the computational results about quadratic semi-deviation, Section 4 derives several useful semivariance formulas for common triangular and trapezoidal fuzzy random variables. Section 5 concludes the paper.

2 Semivariance and Its Properties

Let $(\Omega, \Sigma, \Pr)$ be a probability space. A fuzzy random variable [14] is a map $\xi : \Omega \to \mathcal{F}_\nu$ such that for any Borel subset B of $\Re$, the following function $\mathrm{Pos}\{\xi(\omega) \in B\}$ is measurable with respect to ω, where $\mathcal{F}_\nu$ is a collection of fuzzy variables defined on a possibility space. In addition, the expected value of ξ was defined as (see [14])

$$E[\xi] = \int_0^\infty \Pr\{\omega \in \Omega \mid E[\xi(\omega)] \geq r\}\mathrm{d}r - \int_{-\infty}^0 \Pr\{\omega \in \Omega \mid E[\xi(\omega)] < r\}\mathrm{d}r \quad (1)$$

provided that at least one of the two integrals is finite, where $E[\xi(\omega)]$ is the expected value of fuzzy variable $\xi(\omega)$ (see [20]).

Let ξ be a fuzzy random variable, and its negative part is denoted as $\xi^- = \max\{-\xi, 0\}$, which is also a fuzzy random variable.

Definition 1. *Let ξ be a fuzzy random variable with finite expected value $E[\xi]$. The semivariance of ξ is defined as the expected value of fuzzy random variable $[(\xi - E[\xi])^-]^2$, that is*

$$SV[\xi] = E[(\xi - E[\xi])^-]^2. \tag{2}$$

Similar to the property of variance [14], the semivariance preserves the following property.

Theorem 1. *Let ξ be a fuzzy random variable. Then $SV[\xi] = 0$ if and only if $\xi = E[\xi]$.*

The following theorem deals with the interconnection between variance and semivariance.

Theorem 2. *Let ξ be a fuzzy random variable with finite expected value $E[\xi]$. Then we have*

(i) $SV[\xi] \leq V[\xi]$, *and*
(ii) $SV[\xi] = 0$ *if and if* $V[\xi] = 0$.

3 Quadratic Semideviation of Fuzzy Variable

Let ξ be a triangular fuzzy random variable such that for each ω, $\xi(\omega) = (X(\omega), \alpha, \beta)$ is a triangular fuzzy variable, where $\alpha > 0, \beta > 0$, and X is a random variable with finite expected value. Then for each ω, the expected value of fuzzy variable $\xi(\omega)$ is $E[\xi(\omega)] = (4X(\omega) - \alpha + \beta)/4$. Moreover, by formula (1), we have $E[\xi] = (4E[X] - \alpha + \beta)/4$.

Given a random event ω, we now compute the quadratic semi-deviation of fuzzy variable $E[(\xi(\omega) - E[\xi])^-]^2$. The first result is as follows.

Proposition 1. *Let ξ be a triangular fuzzy random variable such that for each ω, $\xi(\omega) = (X(\omega), \alpha, \beta)$ with $\alpha \neq \beta$, and denote $m = (4E[X] - \alpha + \beta)/4$ as the expected value of ξ.*

(i) If $m + \alpha < X(\omega)$, then $E[(\xi(\omega) - m)^-]^2 = 0$;
(ii) If $m < X(\omega) \leq m + \alpha$, then $E[(\xi(\omega) - m)^-]^2 = -(X(\omega) - \alpha - m)^3/6\alpha$;
(iii) If $m - \beta < X(\omega) \leq m$, then $E[(\xi(\omega) - m)^-]^2 = -(X(\omega) - m)^3/6\beta + (X(\omega) - m)^2/2 - \alpha(X(\omega) - m)/2 + \alpha^2/6$;
(iv) If $X(\omega) \leq m - \beta$, then $E[(\xi(\omega) - m)^-]^2 = (X(\omega) - m)^2 + \beta(X(\omega) - m)/2 - \alpha(X(\omega) - m)/2 + (\alpha^2 + \beta^2)/6$.

Example 1. Let ξ be a triangular fuzzy random variable. For each ω, $\xi(\omega) = (X(\omega) - 3, X(\omega), X(\omega) + 1)$ with $\alpha = 3$, $\beta = 1$, and $X \sim \mathcal{N}(1, 1)$. In this case, the expected value of ξ is $m = 1/2$. By Proposition 1, we have:

(*i*) If $7/2 < X(\omega)$, then $E[(\xi(\omega) - m)^-]^2 = 0$;
(*ii*) If $1/2 < X(\omega) \leq 7/2$, then $E[(\xi(\omega) - m)^-]^2 = -X(\omega)^3/18 + 7X(\omega)^2/12 - 49X(\omega)/24 + 343/144$;

(*iii*) If $-1/2 < X(\omega) \leq 1/2$, then $E[(\xi(\omega) - m)^-]^2 = -X(\omega)^3/6 + 3X(\omega)^2/4 - 17X(\omega)/8 + 115/48$;
(*iv*) If $X(\omega) \leq -1/2$, then $E[(\xi(\omega) - m)^-]^2 = X(\omega)^2 - 2X(\omega) + 29/12$.

Let ξ be a trapezoidal fuzzy random variable such that for each ω, $\xi(\omega) = (X(\omega) - \delta, X(\omega) - \alpha, X(\omega) + \alpha, X(\omega) + \beta)$ is a trapezoidal fuzzy variable, where $\delta > \alpha > 0, \beta > \alpha > 0$, and X is a random variable. Then for each ω, the expected value of $\xi(\omega)$ is $E[\xi(\omega)] = (4X(\omega) - \delta + \beta)/4$. Moreover, by formula (1), we have $E[\xi] = (4E[X] - \delta + \beta)/4$. As a consequence, given a random event ω, we can compute the quadratic semi-deviation of fuzzy variable $E[(\xi(\omega) - E[\xi])^-]^2$, and the computational results are reported in the following proposition.

Proposition 2. *Let ξ be a trapezoidal fuzzy random variable such that for each ω, $\xi(\omega) = (X(\omega) - \delta, X(\omega) - \alpha, X(\omega) + \alpha, X(\omega) + \beta)$ with $\delta \neq \beta$, $\xi > \alpha \geq 0$, $\beta > \alpha \geq 0$, and denote $m = (4E[X] - \delta + \beta)/4$ as the expected value of ξ.*

(*i*) *If $m + \delta \leq X(\omega)$, then $E[(\xi(\omega) - m)^-]^2 = 0$;*
(*ii*) *If $m + \alpha < X(\omega) \leq m + \delta$, then $E[(\xi(\omega) - m)^-]^2 = -(X(\omega) - m - \delta)^3/6(\delta - \alpha)$;*
(*iii*) *If $m - \alpha < X(\omega) \leq m + \alpha$, then $E[(\xi(\omega) - m)^-]^2 = (X(\omega) - m)^2/2 - (\delta + \alpha)(X(\omega) - m)/2 + (\alpha^2 + \alpha\delta + \delta^2)/6$;*
(*iv*) *If $m - \beta < X(\omega) \leq m - \alpha$, then $E[(\xi(\omega) - m)^-]^2 = -(X(\omega) - m + \alpha)^3/6(\beta - \alpha) + (X(\omega) - m)^2/2 - (\delta + \alpha)(X(\omega) - m)/2 + (\alpha^2 + \alpha\delta + \delta^2)/6$;*
(*v*) *If $X(\omega) \leq m - \beta$, then $E[(\xi(\omega) - m)^-]^2 = (X(\omega) - m)^2 - (\delta - \beta)(X(\omega) - m)/2 + (\beta^2 + 2\alpha^2 + \delta^2 + \alpha\beta + \alpha\delta)/6$.*

Example 2. Let ξ be a trapezoidal fuzzy random variable. For each ω, $\xi(\omega) = (X(\omega) - 4, X(\omega) - 1, X(\omega) + 1, X(\omega) + 2)$ with $\delta = 4$, $\alpha = 1$, $\beta = 2$, and $X \sim \mathcal{N}(1, 4)$. In this case, the expected value of ξ is $m = 1/2$. By Proposition 2, we have:

(*i*) If $9/2 < X(\omega)$, then $E[(\xi(\omega) - m)^-]^2 = 0$;
(*ii*) If $3/2 < X(\omega) \leq 9/2$, then $E[(\xi(\omega) - m)^-]^2 = -X(\omega)^3/18 + 3X(\omega)^2/4 - 27X(\omega)/8 + 81/16$;
(*iii*) If $-1/2 < X(\omega) \leq 3/2$, then $E[(\xi(\omega) - m)^-]^2 = X(\omega)^2/2 - 3X(\omega) + 117/24$;
(*iv*) If $-3/2 < X(\omega) \leq -1/2$, then $E[(\xi(\omega) - m)^-]^2 = -X(\omega)^3/6 + X(\omega)^2/4 - 25X(\omega)/8 + 233/48$;
(*v*) If $X(\omega) \leq -3/2$, then $E[(\xi(\omega) - m)^-]^2 = X(\omega)^2 - 2X(\omega) + 65/12$.

4 Semivariance Formulas for Fuzzy Random Variables

In this section, we apply the quadratic semi-deviation formulas derived in Section 3 to frequently used triangular and trapezoidal fuzzy random variables to establish some useful semivariance formulas.

First, for a triangular fuzzy random variable, we have the following result:

Theorem 3. *Let ξ be a triangular fuzzy random variable such that for each ω, $\xi(\omega) = (X(\omega), \alpha, \beta)$ with $\alpha \neq \beta$, and denote $m = (4E[X] - \alpha + \beta)/4$ as the expected value of ξ. If $X \sim \mathcal{N}(\mu, \sigma^2)$, then*

$$\begin{aligned}SV[\xi] = &(\tfrac{5\alpha+\beta}{8\alpha}\sigma^2 + \tfrac{13\alpha^3+13\alpha^2\beta+39\alpha\beta^2-\beta^3}{384\alpha}) + (\tfrac{\sigma^3}{3\sqrt{2\pi}\alpha} + \tfrac{(3\alpha+\beta)^2}{96\sqrt{2\pi}\alpha}\sigma)\exp(-\tfrac{(3\alpha+\beta)^2}{32\sigma^2})\\
&+(\tfrac{\alpha-\beta}{3\sqrt{2\pi}\alpha\beta}\sigma^3 + \tfrac{(\alpha-\beta)^3}{96\sqrt{2\pi}\alpha\beta}\sigma)\exp(-\tfrac{(\alpha-\beta)^2}{32\sigma^2})\\
&-(\tfrac{\sigma^3}{3\sqrt{2\pi}\beta} + \tfrac{\alpha^2+6\alpha\beta-81\beta^2}{96\sqrt{2\pi}\beta}\sigma)\exp(-\tfrac{(\alpha+3\beta)^2}{32\sigma^2})\\
&+(\tfrac{3\alpha+\beta}{8\alpha}\sigma^2 + \tfrac{(3\alpha+\beta)^3}{384\alpha})\Phi(\tfrac{3\alpha+\beta}{4\sigma})\\
&+(\tfrac{(\alpha-\beta)^2}{8\alpha\beta}\sigma^2 + \tfrac{-\alpha^4+58\alpha^3\beta+48\alpha\beta^3+22\alpha^2\beta^2+\beta^4}{384\alpha\beta})\Phi(\tfrac{\alpha-\beta}{4\sigma})\\
&-(\tfrac{\alpha+3\beta}{8\beta}\sigma^2 - \tfrac{\alpha^3-71\alpha^2\beta-69\alpha\beta^2-53\beta^3}{384\beta})\Phi(\tfrac{\alpha+3\beta}{4\sigma}).\end{aligned}$$

Proof. Since $X \sim \mathcal{N}(\mu, \sigma^2)$, we have $E[X] = \mu$, and $m = (4\mu - \alpha + \beta)/4$. By Proposition 1, we have the following computational results.

If $m + \alpha < X(\omega)$, then $SV_1 = 0$. If $m < X(\omega) \leq m + \alpha$, then

$$\begin{aligned}SV_2 &= \int_m^{m+\alpha}(-\tfrac{1}{6\alpha})(x - m - \alpha)^3 \times \tfrac{1}{\sqrt{2\pi}\sigma}\exp(-\tfrac{(x-\mu)^2}{2\sigma^2})dx\\
&= -(\tfrac{(3\alpha+\beta)^3}{384\alpha} + \tfrac{3\alpha+\beta}{8\alpha}\sigma^2) + (\tfrac{\sigma^3}{3\sqrt{2\pi}\alpha} + \tfrac{(3\alpha+\beta)^2}{96\sqrt{2\pi}\alpha}\sigma)\exp(-\tfrac{(3\alpha+\beta)^2}{32\sigma^2})\\
&\quad -(\tfrac{\sigma^3}{3\sqrt{2\pi}\alpha} + \tfrac{37\alpha^2+10\alpha\beta+\beta^2}{96\sqrt{2\pi}\alpha}\sigma)\exp(-\tfrac{(\alpha-\beta)^2}{32\sigma^2})\\
&\quad +(\tfrac{(3\sigma+\beta)}{8\alpha}\sigma^2 + \tfrac{(3\alpha+\beta)^3}{384\alpha})\Phi(\tfrac{3\alpha+\beta}{4\sigma}) + (\tfrac{(3\alpha+\beta)}{8\alpha}\sigma^2 + \tfrac{(3\alpha+\beta)^3}{384\alpha})\Phi(\tfrac{\alpha-\beta}{4\sigma}).\end{aligned}$$

If $m - \beta < X(\omega) \leq m$, then

$$\begin{aligned}SV_3 &= \int_{m-\beta}^{m}[-\tfrac{(x-m)^3}{6\beta} + \tfrac{(x-m)^2}{2} - \tfrac{\alpha(x-m)}{2} + \tfrac{\alpha^2}{6}] \times \tfrac{1}{\sqrt{2\pi}\sigma}\exp(-\tfrac{(x-\mu)^2}{2\sigma^2})dx\\
&= (\tfrac{\sigma^3}{3\sqrt{2\pi}\beta} + \tfrac{\alpha^2+34\alpha\beta+13\beta^2}{96\sqrt{2\pi}\beta}\sigma)\exp(-\tfrac{(\alpha-\beta)^2}{32\sigma^2})\\
&\quad -(\tfrac{\sigma^3}{3\sqrt{2\pi}\beta} + \tfrac{\alpha^2+30\alpha\beta-9\beta^2}{96\sqrt{2\pi}\beta}\sigma)\exp(-\tfrac{(\alpha+3\beta)^2}{32\sigma^2})\\
&\quad +(\tfrac{(\alpha-5\beta)}{8\beta}\sigma^2 - \tfrac{\alpha^3-31\alpha^2\beta-21\alpha\beta^2-13\beta^3}{384\beta}) \times \Phi(\tfrac{\alpha-\beta}{4\sigma})\\
&\quad -(\tfrac{(\alpha-5\beta)}{8\beta}\sigma^2 - \tfrac{\alpha^3+25\alpha^2\beta+27\alpha\beta^2+11\beta^3}{384\beta})\Phi(\tfrac{\alpha+3\beta}{4\sigma}).\end{aligned}$$

If $X(\omega) \leq m - \beta$, then

$$\begin{aligned}SV_4 &= \int_{-\infty}^{m-\beta}[(x - m)^2 + \tfrac{\beta-\alpha}{2}(x - m) + \tfrac{\alpha^2+\beta^2}{6}] \times \tfrac{1}{\sqrt{2\pi}\sigma}\exp(-\tfrac{(x-\mu)^2}{2\sigma^2})dx\\
&= \sigma^2 + \tfrac{5\alpha^2+5\beta^2+6\alpha\beta}{48} + (\tfrac{\alpha+3\beta}{4\sqrt{2\pi}}\sigma)\exp(-\tfrac{(\alpha+3\beta)^2}{32\sigma^2})\\
&\quad -(\sigma^2 + \tfrac{5\alpha^2+5\beta^2+6\alpha\beta}{48})\Phi(\tfrac{\alpha+3\beta}{4\sigma}).\end{aligned}$$

Noting that $SV[\xi] = SV_1 + SV_2 + SV_3 + SV_4$, then the proof of the theorem is complete by combining the above computational results.

We now provide an example to illustrate how to compute the semivariance of a triangular fuzzy random variable.

Example 3. Let ξ be the triangular fuzzy random variable defined in Example 1. For each ω, $\xi(\omega)$ is a triangular fuzzy variable. Since $\alpha = 3$, $\beta = 1$, $\sigma^2 = 1$, and $X \sim \mathcal{N}(1,1)$, it follows from Theorem 3 that $SV[\xi] = 8.99$.

Furthermore, for a trapezoidal fuzzy random variable, we have

Theorem 4. *Let ξ be a trapezoidal fuzzy random variable such that for each ω, $\xi(\omega) = (X(\omega) - \delta, X(\omega) - \alpha, X(\omega) + \alpha, X(\omega) + \beta)$ with $\delta \neq \beta$, $\xi > \alpha \geq 0$, $\beta > \alpha \geq 0$, and denote $m = (4E[X] - \delta + \beta)/4$ as the expected value of ξ. If $X \sim \mathcal{N}(\mu, \sigma^2)$, then*

$$
\begin{aligned}
SV[\xi] = {} & (\tfrac{1}{2}\sigma^2 + \tfrac{16\alpha^2+7\beta^2+3\delta^2+4\alpha\beta+12\alpha\delta+6\beta\delta}{96}) \\
& +(\tfrac{\sigma^3}{3\sqrt{2\pi}(\delta-\alpha)} + \tfrac{(3\delta+\beta)^2}{96\sqrt{2\pi}(\delta-\alpha)}\sigma)\exp(-\tfrac{(3\delta+\beta)^2}{32\sigma^2}) \\
& -(\tfrac{\sigma^3}{3\sqrt{2\pi}(\delta-\alpha)} + \tfrac{16\alpha^2+\beta^2+\delta^2+8\alpha\beta-8\alpha\delta-2\beta\delta}{96\sqrt{2\pi}(\delta-\alpha)}\sigma)\exp(-\tfrac{(4\alpha-\delta+\beta)^2}{32\sigma^2}) \\
& +(\tfrac{\sigma^3}{3\sqrt{2\pi}(\beta-\alpha)} + \tfrac{16\alpha^2+\beta^2+\delta^2-8\alpha\beta+8\alpha\delta-2\beta\delta}{96\sqrt{2\pi}(\beta-\alpha)}\sigma)\exp(-\tfrac{(4\alpha+\delta-\beta)^2}{32\sigma^2}) \\
& -(\tfrac{\sigma^3}{3\sqrt{2\pi}(\beta-\alpha)} + \tfrac{(\delta+3\beta)^2}{96\sqrt{2\pi}(\beta-\alpha)}\sigma)\exp(-\tfrac{(\delta+3\beta)^2}{32\sigma^2}) + (\tfrac{(3\delta+\beta)^3}{384(\delta-\alpha)} + \tfrac{3\delta+\beta}{8(\delta-\alpha)}\sigma^2)\Phi(\tfrac{3\delta+\beta}{4\sigma}) \\
& +(\tfrac{1}{2}\sigma^2 + \tfrac{16\alpha^2+3\beta^2+7\delta^2+12\alpha\beta+4\alpha\delta+6\beta\delta}{96} - \tfrac{(3\delta+\beta)^3}{384(\delta-\alpha)} - \tfrac{3\delta+\beta}{8(\delta-\alpha)}\sigma^2)\Phi(\tfrac{4\alpha-\delta+\beta}{4\sigma}) \\
& +(\tfrac{1}{2}\sigma^2 + \tfrac{(4\alpha+\delta-\beta)^3}{384(\beta-\alpha)} + \tfrac{8\alpha+\delta-5\beta}{8(\beta-\alpha)}\sigma^2)\Phi(\tfrac{4\alpha+\delta-\beta}{4\sigma}) \\
& -(\sigma^2 + \tfrac{(4\alpha+\delta-\beta)^3}{384(\beta-\alpha)} + \tfrac{8\alpha+\delta-5\beta}{8(\beta-\alpha)}\sigma^2 + \tfrac{16\alpha^2+7\beta^2+3\delta^2+4\alpha\beta+12\alpha\delta+6\beta\delta}{96})\Phi(\tfrac{\delta+3\beta}{4\sigma}).
\end{aligned}
$$

Proof. Since $X \sim \mathcal{N}(\mu, \sigma^2)$, we have $m = (4\mu - \delta + \beta)/4$. By using Proposition 2, we can derive the semivariance of ξ. The computational process is divided into several steps.

Step I. If $m + \delta < X(\omega)$, then $SV_1 = 0$.

Step II. If $m + \alpha < X(\omega) \leq m + \alpha$, then

$$
\begin{aligned}
SV_2 = {} & \textstyle\int_{m+\alpha}^{m+\delta}[-(x-m-\delta)^3/6(\delta-\alpha)]\frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{(x-\mu)^2}{2\sigma^2})dx \\
= {} & (\tfrac{\sigma^3}{3\sqrt{2\pi}(\delta-\alpha)} + \tfrac{(3\delta+\beta)^2}{96\sqrt{2\pi}(\delta-\alpha)}\sigma)\exp(-\tfrac{(3\delta+\beta)^2}{32\sigma^2}) \\
& -(\tfrac{\sigma^3}{3\sqrt{2\pi}(\delta-\alpha)} + \tfrac{16\alpha^2+\beta^2+37\delta^2-4\alpha\beta-44\alpha\delta+10\beta\delta}{96\sqrt{2\pi}(\delta-\alpha)}\sigma)\exp(-\tfrac{(4\alpha-\delta+\beta)^2}{32\sigma^2}) \\
& +(\tfrac{(3\delta+\beta)^3}{384(\delta-\alpha)} + \tfrac{(3\delta+\beta)}{8(\delta-\alpha)}\sigma^2)\Phi(\tfrac{3\delta+\beta}{4\sigma}) - (\tfrac{(3\delta+\beta)^3}{384(\delta-\alpha)} + \tfrac{(3\delta+\beta)}{8(\delta-\alpha)}\sigma^2)\Phi(\tfrac{4\alpha-\delta+\beta}{4\sigma}).
\end{aligned}
$$

Step III. If $m - \alpha < X(\omega) \leq m + \alpha$, then

$$
\begin{aligned}
SV_3 = {} & \textstyle\int_{m-\alpha}^{m+\alpha}[\frac{1}{2}(x-m)^2 - \frac{1}{2}(\delta+\alpha)(x-m) \\
& +\frac{1}{6}(\delta^2+\alpha\delta+\alpha^2)] \times \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{(x-\mu)^2}{2\sigma^2})dx \\
= {} & -\tfrac{\sigma^2}{2} - \tfrac{16\alpha^2+3\beta^2+7\delta^2+12\alpha\beta+4\alpha\delta+6\beta\delta}{96} + \tfrac{3\delta+\beta}{8\sqrt{2\pi}}\sigma\exp(-\tfrac{(4\alpha-\delta+\beta)^2}{32\sigma^2}) \\
& -\tfrac{8\alpha+3\delta+\beta}{8\sqrt{2\pi}}\sigma\exp(-\tfrac{(4\alpha+\delta-\beta)^2}{32\sigma^2}) \\
& +(\tfrac{\sigma^2}{2} + \tfrac{16\alpha^2+3\beta^2+7\delta^2+12\alpha\beta+4\alpha\delta+6\beta\delta}{96})\Phi(\tfrac{4\alpha-\delta+\beta}{4\sigma}) \\
& +(\tfrac{\sigma^2}{2} + \tfrac{16\alpha^2+3\beta^2+7\delta^2+12\alpha\beta+4\alpha\delta+6\beta\delta}{96})\Phi(\tfrac{4\alpha+\delta-\beta}{4\sigma}).
\end{aligned}
$$

Step IV. If $m-\beta < X(\omega) \le m-\alpha$, then

$$\begin{aligned}
SV_4 &= \int_{m-\beta}^{m-\alpha}[-\tfrac{1}{6(\beta-\alpha)}(x-m+\alpha)^3+\tfrac{1}{2}(x-m)^2 \\
&\quad -\tfrac{1}{2}(\delta+\alpha)(x-m)+\tfrac{1}{6}(\alpha^2+\alpha\delta+\delta^2)]\times\tfrac{1}{\sqrt{2\pi}\sigma}\exp(-\tfrac{(x-\mu)^2}{2\sigma^2})dx \\
&= (\tfrac{\sigma^3}{3\sqrt{2\pi}(\beta-\alpha)}+\tfrac{-80\alpha^2+13\beta^2+\delta^2+76\alpha\beta-28\alpha\delta+34\beta\delta}{96\sqrt{2\pi}(\beta-\alpha)}\sigma)\exp(-\tfrac{(4\alpha+\delta-\beta)^2}{32\sigma^2}) \\
&\quad -(\tfrac{\sigma^3}{3\sqrt{2\pi}(\beta-\alpha)}+\tfrac{-81\beta^2+\delta^2-72\alpha\beta-24\alpha\delta+30\beta\delta}{96\sqrt{2\pi}(\beta-\alpha)}\sigma)\exp(-\tfrac{(\delta+3\beta)^2}{32\sigma^2}) \\
&\quad +(\tfrac{(4\alpha+\delta-\beta)^3}{384(\beta-\alpha)}+\tfrac{8\alpha+\delta-5\beta}{8(\beta-\alpha)}\sigma^2-\tfrac{16\alpha^2+3\beta^2+7\delta^2+12\alpha\beta+4\alpha\delta+6\beta\delta}{96})\Phi(\tfrac{4\alpha+\delta-\beta}{4\sigma}) \\
&\quad -(\tfrac{(4\alpha+\delta-\beta)^3}{384(\beta-\alpha)}+\tfrac{8\alpha+\delta-5\beta}{8(\beta-\alpha)}\sigma^2-\tfrac{16\alpha^2+3\beta^2+7\delta^2+12\alpha\beta+4\alpha\delta+6\beta\delta}{96})\Phi(\tfrac{\delta+3\beta}{4\sigma}).
\end{aligned}$$

Step V. If $X(\omega) \le m-\beta$, then

$$\begin{aligned}
SV_5 &= \int_{-\infty}^{m-\beta}[(x-m)^2-\tfrac{1}{2}(\delta-\beta)(x-m) \\
&\quad +\tfrac{1}{6}(\beta^2+2\alpha^2+\delta^2+\alpha\beta+\alpha\delta)]\times\tfrac{1}{\sqrt{2\pi}\sigma}\exp(-\tfrac{(x-\mu)^2}{2\sigma^2})dx \\
&= (\sigma^2+\tfrac{16\alpha^2+5\beta^2+5\delta^2+8\alpha\beta+8\alpha\delta+6\beta\delta}{48})+\tfrac{\delta+3\beta}{4\sqrt{2\pi}}\sigma\exp(-\tfrac{(\delta+3\beta)^2}{32\sigma^2}) \\
&\quad -(\sigma^2+\tfrac{16\alpha^2+5\beta^2+5\delta^2+8\alpha\beta+8\alpha\delta+6\beta\delta}{48})\Phi(\tfrac{\delta+3\beta}{4\sigma}).
\end{aligned}$$

Since $SV[\xi] = SV_1 + SV_2 + SV_3 + SV_4 + SV_5$, it follows from Step I to Step V that the proof of the theorem is complete.

As an application of Theorem 4, consider the following example:

Example 4. Let ξ be the trapezoidal fuzzy random variable defined in Example 2. Noting that for each ω, $\xi(\omega)$ is a trapezoidal fuzzy variable, and $X \sim \mathcal{N}(1,4)$, then Theorem 4 implies $SV[\xi] = 3.91$.

5 Conclusions and Future Research

To model the choice among uncertain outcomes, mean-risk approach is frequently used in the literature. This paper presented a new risk criteria for the measure of risk in hybrid uncertain systems. The major new results include the following several aspects. (i) The semivariance of a fuzzy random variable was defined as a new risk measure for hybrid uncertain outcomes, and the fundamental properties of the semivariance were analyzed (Theorems 1 and 2); (ii) Given a random outcome, the formulas for the quadratic semi-deviation of fuzzy variable were established (Propositions 1 and 2), and (iii) the semivariance formulas for frequently used triangular and trapezoidal fuzzy random variables were derived (Theorem 3 and 4).

The established formulas about quadratic semi-deviation and semivariance can be used in various practical risk management problems, which will be our future research.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (NSFC) under grant No. 60974134.

References

1. Markowitz, H.M.: Portfolio Selection. J. Finan. 7, 77–91 (1952)
2. Yan, W., Miao, R., Li, S.: Mutiperiod Semivariance Portfolio Selection: Model and Numerical Solution. Appl. Math. Comput. 194, 128–134 (2007)
3. Ogryczak, W., Ruszczyński, A.: From Stochastic Dominance to Mean Risk Models: Semideviations as Risk Measures. Eur. J. Oper. Res. 116, 33–50 (1999)
4. Wagnei, M., Bhadury, J., Peng, S.: Risk Management in Uncapcitated Facility Location Models with Random Demands. Comput. Oper. Res. 36, 1002–1011 (2009)
5. Chang, T., Yang, S., Chang, K.: Portfolio Optimization Problems in Different Risk Measures Using Genetic Algorithm. Expert. Syst. Appl. 36, 10529–10537 (2009)
6. Graves, S., Ringuest, J.: Probabilistic Dominance Criteria for Comparing Uncertain Alternatives: a Tutorial. Omega 37, 346–357 (2009)
7. Kwakernaak, H.: Fuzzy Random Variables-I. Inform. Sci. 15, 1–29 (1978)
8. Feng, X., Liu, Y.: Measurability Criteria for Fuzzy Random Vectors. Fuzzy Optim. Decis. Mak. 5, 245–253 (2006)
9. Hao, F.F., Liu, Y.K.: Mean Variance Models for Portfolio Selection with Fuzzy Random Returns. J. Appl. Math. Comput. 30, 9–38 (2009)
10. Hao, F.F., Qin, R.: Variance Formulas for Trapezoidal Fuzzy Random Variables. J. Uncertain Syst. 2, 145–160 (2009)
11. Liu, B.: Uncertainty Theory. Springer, Berlin (2004)
12. Liu, Y.: The Approximation Method for Two Stage Fuzzy Random Programming with Recourse. IEEE Trans. Fuzzy Syst. 15, 1197–1208 (2007)
13. Liu, Y.K.: The Convergent Results about Approximating Fuzzy Random Minimum Risk Problems. Appl. Math. Comput. 205, 608–621 (2008)
14. Liu, Y.K., Liu, B.: Fuzzy Random Variable: a Scalar Expected Value Operator. Fuzzy Optim. Decis. Mak. 2, 143–160 (2003)
15. Liu, Y.K., Liu, Z.Q., Gao, J.: The Modes of Convergence in the Approximation of Fuzzy Random Optimization Problems. Soft Comput. 13, 117–125 (2009)
16. Qin, R., Hao, F.: Computing the Mean Chance Distributions of Fuzzy Random Variables. J. Uncertain Syst. 2, 299–312 (2008)
17. Wang, S., Liu, Y.K., Watada, J.: Fuzzy Random Renewal Process with Queueing Applications. Comput. Math. Appl. 57, 1232–1248 (2009)
18. Markowitz, H.: Portfolio Selection: Efficient Diversification of Investments. Wiley, New York (1959)
19. Estrada, J.: Mean Semivariance Behavior: Downside Risk and Capital Asset Pricing. Int. Review Econ. Finan. 16, 169–185 (2007)
20. Liu, B., Liu, Y.K.: Expected Value of Fuzzy Variable and Fuzzy Expected Value Models. IEEE Trans. Fuzzy Syst. 10, 445–450 (2002)

Enhanced Extreme Learning Machine with Modified Gram-Schmidt Algorithm

Jianchuan Yin[1] and Nini Wang[2]

[1] College of Navigation, Dalian Maritime University,
1 Linghai Road, Dalian 116026, China
yinjianchuan@gmail.com
[2] Department of Mathematics, Dalian Maritime University,
1 Linghai Road, Dalian 116026, China
wangnini@dlmu.edu.cn

Abstract. Extreme learning machine (ELM) has shown to be extremely fast with better generalization performance. However, the implementation of ELM encounters two problems. First, ELM tends to require more hidden nodes than conventional tuning-based algorithms. Second, subjectivity is involved in choosing hidden nodes number. In this paper, we apply the modified Gram-Schmidt (MGS) method to select hidden nodes which maximize the increment to explained variance of the desired output. The Akaike's final prediction error (FPE) criterion are used to automatically determine the number of hidden nodes. In comparison with conventional ELM learning method on several commonly used regressor benchmark problems, our proposed algorithm can achieve compact network with much faster response and satisfactory accuracy.

Key words: Extreme learning machine (ELM), Modified Gram-Schmidt algorithm (MGS), Feedforward neural networks.

1 Introduction

Huang et al. have recently proposed a new theory to show that SLFNs with randomly generated additive or RBF hidden nodes can work as universal approximators [1,2]. Strongly supported by this theoretical results, Huang et al. have proposed a new learning algorithm for the feedforward neural network referred to as extreme learning machine (ELM). In ELM, the parameters of hidden nodes are randomly selected and the output weights are analytically determined through simple generalized inverse operation of the hidden layer output matrices. The performance of ELM has been evaluated on a number of benchmark problems. In many real world applications, ELM has been shown to generate good generalization performance at extremely high learning speed.

Nevertheless, the implementation of ELM faces two problems. First, ELM tends to require more hidden nodes than conventional tuning-based algorithms. Some of the hidden nodes in such a network may play a much minor role in network output and thus may eventually increase network complexity [3,4,5].

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 381–388, 2010.

Second, when using ELM to handle the problem in hand, it remains a trial and error process. The subjectivity is involved in choosing the number of hidden nodes.

In this paper, we adopt a systematic approach to address these two problems. The selection of hidden nodes in random hidden nodes pool can be regarded as a problem of subset model selection in a linear regression model. The modified Gram-Schmidt (MGS) [6,7,8] algorithm can be employed as a forward regression procedure to select a suitable set of hidden nodes from a large set of candidates. We apply the Akaike's final prediction error (FPE) criterion [9,10] to select the appropriate number of hidden nodes. Various benchmark regressor datasets were used to demonstrate the effectiveness of our proposed algorithm.

2 A Brief Review of ELM

Assume we have N arbitrary distinct samples $(\mathbf{x}_k, \mathbf{t}_k) \in \mathbf{R}^n \times \mathbf{R}^m$, where $\mathbf{x}_k$ is an input vector and $\mathbf{t}_k$ is the corresponding desired output. A standard single hidden layer feedforward networks (SLFNs) with $\tilde{N}$ additive hidden nodes and activation function $G(\mathbf{x})$ can be represented by

$$f_{\tilde{N}}(\mathbf{x}_k) = \sum_{i=1}^{\tilde{N}} \boldsymbol{\omega}_i G(\mathbf{a}_i \cdot \mathbf{x}_k + b_i), \quad k = 1, \ldots, N. \tag{1}$$

where $\mathbf{a}_i = [a_{1i}, \cdots, a_{ni}]^T$ is the weight vector connecting the input layer to the ith hidden node, b_i is the bias of the ith hidden node, and $\mathbf{a}_i \cdot \mathbf{x}_k$ denotes the inner product of vectors $\mathbf{a}_i$ and $\mathbf{x}_k$ in $\mathbf{R}^n$. The activation functions $G(\mathbf{x})$ are sigmoids.

The ultimate purpose of SLFNs is to find out the values of $\omega_i, \mathbf{a}_i$ and b_i such that $\sum_{k=1}^{N} \|f_{\tilde{N}}(\mathbf{x}_k) - t_k\| = 0$, or

$$f_{\tilde{N}}(\mathbf{x}_k) = \sum_{i=1}^{\tilde{N}} \omega_i G(\mathbf{a}_i \cdot \mathbf{x}_k + b_i) = t_k, \quad k = 1, \ldots, N. \tag{2}$$

Then, (2) can be written compactly as

$$\mathbf{H}\boldsymbol{\omega} = \mathbf{T}, \tag{3}$$

where

$$\begin{aligned} &\mathbf{H}(\mathbf{a}_1, \ldots, \mathbf{a}_{\tilde{N}}, b_1, \ldots, b_{\tilde{N}}, \mathbf{x}_1, \ldots, \mathbf{x}_N) \\ &= \begin{bmatrix} G(\mathbf{a}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & G(\mathbf{a}_{\tilde{N}} \cdot \mathbf{x}_1 + b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ G(\mathbf{a}_1 \cdot \mathbf{x}_N + b_1) & \cdots & G(\mathbf{a}_{\tilde{N}} \cdot \mathbf{x}_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}}, \end{aligned} \tag{4}$$

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_{\tilde{N}} \end{bmatrix} \text{ and } \quad \mathbf{T} = \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix}. \tag{5}$$

The main idea of ELM [1] is that for N arbitrary distinct samples $(\mathbf{x}_k, t_k)$ in order to obtain arbitrarily small non-zero training error, one may randomly generate $\tilde{N}(\leq N)$ hidden nodes (with random parameters $\mathbf{a}_i$ and b_i). Under this assumption, $\mathbf{H}$ is completely defined. Then, (3) becomes a linear system and the output weights $\boldsymbol{\omega}$ are estimated as

$$\hat{\boldsymbol{\omega}} = \mathbf{H}^{\dagger}\mathbf{T}, \tag{6}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of the hidden layer output matrix $\mathbf{H}$.

3 Enhanced-ELM

An efficient learning procedure for selecting suitable random hidden nodes can readily be derived on the modified Gram-Schmidt (MGS) method [6,7,8,10]. In order to understand the proposed algorithm, it is essential to view the network (1) as a special case of the linear regression model

$$t_k = \sum_{i=1}^{\tilde{N}} \omega_i G(\mathbf{a}_i \cdot \mathbf{x}_k + b_i) + \epsilon_k, \quad k = 1, \ldots, N. \tag{7}$$

This construction procedure can be described by the following matrix equation:

$$\mathbf{T} = \mathbf{H}\boldsymbol{\omega} + \mathbf{E}, \tag{8}$$

where $\mathbf{E} = [\epsilon_1, \cdots, \epsilon_N]^T$.

Then, let us define a set of vectors $\boldsymbol{\psi}_i (i = 1, \cdots, \tilde{N})$ by

$$\boldsymbol{\psi}_i = \zeta_i \begin{bmatrix} G(\mathbf{a}_i \cdot \mathbf{x}_1 + b_i) \\ \vdots \\ G(\mathbf{a}_i \cdot \mathbf{x}_N + b_i) \end{bmatrix}, \quad i = 1, \cdots, \tilde{N}, \tag{9}$$

where $\boldsymbol{\psi}_i$ is, in fact, the normalized version of the column-vectors of $\mathbf{H}$ in (8), and ζ_i satisfies

$$\zeta_i = (\sum_{k=1}^{N} (G(\mathbf{a}_i \cdot \mathbf{x}_k + b_i))^2)^{-\frac{1}{2}}, \quad i = 1, \cdots, \tilde{N}, \tag{10}$$

so that $\boldsymbol{\psi}_i^T \boldsymbol{\psi}_i = 1, i = 1, \cdots, \tilde{N}$.

So a modified equation consists in replacing (8) by

$$\mathbf{T} = \mathbf{H}_0\boldsymbol{\omega} + \mathbf{E}, \tag{11}$$

where $\mathbf{H}_0 = [\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_{\tilde{N}}]$.

The problem of how to select a suitable set of hidden nodes from the random hidden nodes pool can be regarded as an example of how to select a subset of

significant regressors from the given candidate set $\{\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_{\tilde{N}}\}$ in a forward-regression manner.

At iteration m, denote by l_m the index of the hidden node selected from random hidden nodes pool. In other words, $\boldsymbol{\psi}_{l_m}$ is selected at iteration m. Let m be the current iteration number, then $\boldsymbol{\psi}_{l_1}, \boldsymbol{\psi}_{l_2}, \cdots, \boldsymbol{\psi}_{l_{m-1}}$ have been selected in the previous iterations. Define the intermediate vectors by

$$\boldsymbol{p}_{l_j}(i) = \boldsymbol{p}_{l_j}(i-1) - (\boldsymbol{\psi}_{l_j}^T \boldsymbol{q}_{l_{i-1}})\boldsymbol{q}_{l_{i-1}}, i = 1, \cdots, m-1, j = i, \cdots, m-1, \quad (12)$$

with $\boldsymbol{p}_{l_j}(0) = \boldsymbol{\psi}_{l_j}$ and $\boldsymbol{q}_{l_0} = 0$. Thus we generate the vectors $\boldsymbol{q}_{l_1}, \cdots, \boldsymbol{q}_{l_{m-1}}$ by

$$\boldsymbol{q}_{l_j} = \left((\boldsymbol{p}_{l_j}(i))^T \boldsymbol{p}_{l_j}(i)\right)^{-\frac{1}{2}} \boldsymbol{p}_{l_j}(i), i = 1, \cdots, m-1, j = i, \cdots, m-1, \quad (13)$$

then $\boldsymbol{q}_{l_j}$ is an orthonormalized version of $\boldsymbol{\psi}_{l_j}, j = 1, \cdots, m-1$.

Now, orthogonalize the remaining vectors $\boldsymbol{\psi}_j$ to $\boldsymbol{q}_{l_1}, \cdots, \boldsymbol{q}_{l_{m-1}}$,

$$\boldsymbol{p}_j(m) = \boldsymbol{p}_j(m-1) - (\boldsymbol{\psi}_j^T \boldsymbol{q}_{l_{m-1}})\boldsymbol{q}_{l_{m-1}}, \quad (14)$$

The vectors $\boldsymbol{\psi}_{l_1}, \boldsymbol{\psi}_{l_2}, \cdots, \boldsymbol{\psi}_{l_{m-1}}$ and $\boldsymbol{\psi}_j$ span the same space as $\boldsymbol{q}_{l_1}, \cdots, \boldsymbol{q}_{l_{m-1}}$, and $\boldsymbol{p}_j(m)$. Since $\boldsymbol{q}_{l_1}, \cdots, \boldsymbol{q}_{l_{m-1}}$, and $\boldsymbol{p}_j(m)$ are orthogonal, the best $\boldsymbol{\psi}_j$ to be chosen for the current iteration corresponds to the $\boldsymbol{p}_j(m)$ the "closest" to $\mathbf{T}$. Thus we should choose

$$l_m = \arg\max_j \frac{(\boldsymbol{p}_j(m))^T \mathbf{T}}{\left((\boldsymbol{p}_j(m))^T \boldsymbol{p}_j(m)\right)^{\frac{1}{2}}}, \quad (15)$$

and accordingly

$$\boldsymbol{q}_{l_m} = \left((\boldsymbol{p}_{l_m}(m))^T \boldsymbol{p}_{l_m}(m)\right)^{-\frac{1}{2}} \boldsymbol{p}_{l_m}(m). \quad (16)$$

The number of hidden nodes can be determined by the Akaike's final prediction error criterion (FPE) [9,10]. The FPE criteria can be written as

$$FPE(j) = \frac{1 + n_j/N}{1 - n_j/N} \sigma_j^2, \quad (17)$$

where n_j is the number of the parameters in the estimator, N is the sample length of the training data, and σ_j^2 is the variance of the residual corresponding to the random hidden nodes selection procedure. This kind of information theoretic criterion is combined with random hidden nodes selection procedure in such a way that regressors are selected based on (15) as described previously and the selection procedure is automatically terminated when $FPE(j)$ reaches its minimum.

From the MGS theory, it is known that

$$[\boldsymbol{\psi}_{l_1}, \ldots, \boldsymbol{\psi}_{l_M}] = [\boldsymbol{q}_{l_1}, \ldots, \boldsymbol{q}_{l_M}]\mathbf{A}, \quad (18)$$

where $\boldsymbol{q}_{l_j}, (j = 1, \cdots, M)$, are orthogonal vectors, which satisfy

$$[\boldsymbol{q}_{l_1}, \ldots, \boldsymbol{q}_{l_M}]^T [\boldsymbol{q}_{l_1}, \ldots, \boldsymbol{q}_{l_M}] = \mathbf{I}, \quad (19)$$

and $\mathbf{A}$ is an upper-triangular matrix, given as

$$\mathbf{A} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \cdots & \alpha_{1M-1} & \alpha_{1M} \\ 0 & \alpha_{22} & \alpha_{23} & \cdots & \alpha_{2M-1} & \alpha_{2M} \\ 0 & 0 & \alpha_{33} & \cdots & \alpha_{3M-1} & \alpha_{3M} \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \alpha_{M-1M-1} & \alpha_{M-1M} \\ 0 & 0 & \cdots & \cdots & 0 & \alpha_{MM} \end{bmatrix}. \tag{20}$$

Hence, the coefficients ω_{l_i} are determined by

$$\begin{bmatrix} \omega_{l_1} \\ \vdots \\ \omega_{l_M} \end{bmatrix} = \mathbf{A}^{\dagger} \begin{bmatrix} \tilde{\omega}_{l_1} \\ \vdots \\ \tilde{\omega}_{l_M} \end{bmatrix} \tag{21}$$

where $\mathbf{A}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{A}$.

The network is then constructed as

$$f_M(\mathbf{x}) = \sum_{i=1}^{M} \omega_{l_i} \boldsymbol{\psi}_{l_i}(\mathbf{x}). \tag{22}$$

4 Performance Evaluation

In [1], authors have demonstrated that ELM can outperform many popular algorithms like BP, SVM and other well-known algorithms in many cases, thus we may only need to compare the performance of the proposed algorithm with the conventional ELM algorithm in this paper.

In this paper, all the simulations have been conducted in MATLAB 7.1 environment running on ordinary PC with 2.0 GHZ CPU and 512M RAM. For simplicity, all the input and output data were normalized into the range [-1,1]. The activation function used in our experiments was a simple sigmoidal additive activation function, the input weights $\mathbf{a}$ were randomly chosen from the range [-1,1] while the biases b were randomly chosen from the range [0,1]. In our study, we considered the number of hidden nodes in the range of [10, 500] and a step size of 10 nodes. This defines the overall training time of the conventional ELM.

4.1 Artificial Case: Approximation of 'SinC' Function with Noise

In this example, we presented the performance of the proposed algorithm and conventional ELM on the 'SinC' function, a popular choice to illustrate ELM in the literature [1]

$$y(x) = \begin{cases} sin(x)/x, & x \neq 0, \\ 1, & x = 0. \end{cases}$$

Both algorithms were used to approximate a training set (x_i, y_i) and testing set (x_i, y_i) with 5000 data, respectively, where x_i's were uniformly randomly

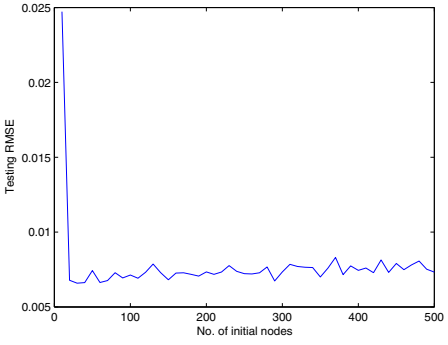


Fig. 1. Performance

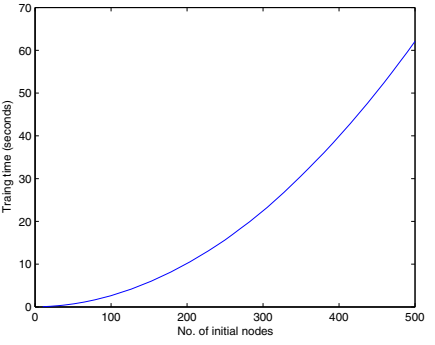


Fig. 2. Training time

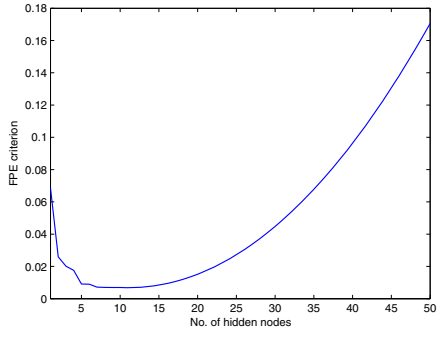


Fig. 3. FPE criterion

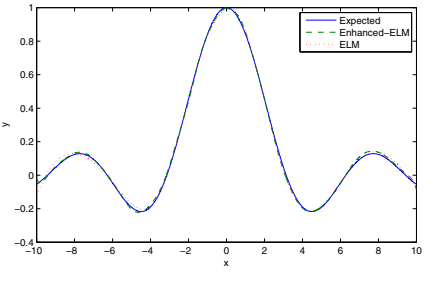


Fig. 4. Outputs

distributed on the interval $[-10, 10]$. In order to make the regression problem 'real', large uniform noise distributed in $[-0.2, 0.2]$ has been added to all the training samples while testing data remain noise-free.

For our proposed algorithm, We conducted an experimental study by varying the number of initial hidden neurons from 10 to 500 in steps of 10. The variation of the testing accuracies and the training time are illustrated in Fig. 1 and Fig. 2. The results indicate 500 initial hidden nodes appears to be sufficient for the problems considered.Fig. 3 shows the FPE values for each optimal hidden nodes number. It suggests to choose 12 hidden nodes for constructing the ELM network.

As shown in Table 1, we compared the performance of the proposed method with ELM approach. It can be seen from Table 1 that enhanced-ELM learning algorithm spent 62.1187s CPU time obtaining the testing root mean square error (RMSE) 0.0073, however, it takes 160.1541s CPU time for conventional ELM algorithm to reach a higher testing error 0.0074. Based on the results, enhanced-ELM arrives at competitive prediction accuracy at the cost of significantly lower training time. Furthermore, enhanced-ELM produces a more compact network compared with conventional ELM algorithm.

Table 1. Performance comparison for learning noise free function: SinC

Algorithm	No. of nodes	Testing		Training times(s)
		RMS	Dev	
enhanced-ELM	12	0.0073	0.0016	62.1187
ELM	20	0.0074	1.6163e-004	160.1541

In Fig. 4, we plotted the true and the approximated function of the enhanced-ELM and conventional ELM learning algorithm. The comparison with the conventional ELM method confirms the more general applicability of our proposed algorithm.

4.2 Benchmarking with Regression Problem

The performance of ELM and our proposed algorithm were compared on five real world benchmark regression datasets from UCI ML repository. The specifications of the data sets are listed in Table 2. For each problem, 50 trials were done.

The performance results are tabulated in Tables 3. In comparison to ELM, enhanced-ELM arrives at competitive prediction accuracy at significantly lower training time. The training time of enhanced-ELM excludes the computational time used to choose the appropriate hidden nodes for the problems considered. Note that what is reported in Table 3, ELM takes a step size of 10 nodes

Table 2. Specification of five benchmark regression datasets

Data set	No. of observations		Attribute
	Training data	Testing data	
Ailerons	7154	6596	40
Bank domains	4500	3692	32
Elevators	8752	7846	18
Kinematics	4000	4192	8
Puma32H	4500	3692	32

Table 3. Performance comparison in more benchmark applications

Data sets	Methods	No. of nodes	Testing		Training time(s)
			RMS	Dev	
Ailerons	enhanced-ELM	90	1.6208e-004	1.1527e-006	94.5750
	ELM	130	1.6173e-004	8.2616e-007	236.1891
Bank domains	enhanced-ELM	71	0.0874	5.2811e-004	60.5031
	ELM	350	0.0877	6.4680e-004	194.6844
Elevators	enhanced-ELM	244	0.0024	2.1214e-004	116.1238
	ELM	250	0.0023	5.7885e-005	297.6750
Kinematics	enhanced-ELM	251	0.1127	0.0025	57.8025
	ELM	500	0.1108	0.0037	182.3547
Puma32H	enhanced-ELM	71	0.0273	1.0047e-004	63.9269
	ELM	90	0.0271	8.7990e-005	171.7672

and enhanced-ELM takes a step size of 1 node. In addition, results illustrate enhanced-ELM produces significantly more compact networks.

5 Conclusions

An enhanced-ELM algorithm is proposed which is more constructive in the sense that it automatically determines the number of hidden nodes and selects the most suitable hidden nodes in random nodes pool. In contrast to conventional ELM algorithm, enhanced-ELM produces compact network structure and generates competitive testing accuracy at cost of significantly lower training time.

References

1. Huang, G.-B., Zhu, Q.-Y., Siew, C.-K.: Extreme learning machine: theory and applications. Neurocomputing 70, 489–501 (2006)
2. Liang, N.-Y., Huang, G.-B., Saratchandran, P., Sundararajan, N.: A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans. Neural Netw. 17, 1411–1423 (2006)
3. Huang, G.-B., Chen, L., Siew, C.-K.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans. Neural Netw. 17, 879–892 (2006)
4. Huang, G.-B., Chen, L.: Enhanced random search based incremental extreme learning machine. Neurocomputing 71, 3460–3468 (2008)
5. Feng, G., Huang, G.-B., Lin, Q., Gay, R.: Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans. Neural Netw. 20, 1352–1357 (2009)
6. Chen, S., Cowan, C.F.N., Grant, P.M.: Orthogonal least squares learning algorithm for radial basis function networks. IEEE Trans. Neural Netw. 2, 302–309 (1991)
7. Chen, S., Billings, S.A., Luo, W.: Orthogonal least squares methods and their application to non-linear system identification. Int. J. Control 50, 1873–1896 (1989)
8. Li, K., Peng, J.X., Irwin, G.W.: A fast nonlinear model identification method. IEEE Trans. Automatic Control 50, 1211–1216 (2005)
9. Akaike, H.: Statistical predictor identification. Ann. Inst. Stat. Math. 22, 202–217 (1970)
10. Zhang, Q.: Using wavelet network in nonparametric estimation. IEEE Trans. Neural Netw. 8, 227–236 (1997)

Solving Large N-Bit Parity Problems with the Evolutionary ANN Ensemble

Lin-Yu Tseng[1,2] and Wen-Ching Chen[2,3]

[1] Institute of Networking and Multimedia,
National Chung Hsing University, Taichung, Taiwan 402, ROC
[2] Department of Computer Science and Engineering,
National Chung Hsing University, Taichung, Taiwan 402, ROC
[3] Department of Information Networking Technology,
Hsiuping Institute of Technology, Taichung, Taiwan, 412, ROC
{lytseng,phd9102}@cs.nchu.edu.tw

Abstract. Artificial neural networks (ANNs) have been successfully applied to many areas due to its powerful ability both for classification and regression problems. For some difficult problems, ANN ensemble classifiers are considered, instead of a single ANN classifier. In the previous study, the authors presented the systematic trajectory search algorithm (STSA) to train the ANN. The STSA utilizes the orthogonal array (OA) to uniformly generate the initial population to globally explore the solution space, and then applies a novel trajectory search method to exploit the promising areas thoroughly. In this paper, an evolutionary constructing algorithm, called the ESTSA, of the ANN ensemble is proposed. Based on the STSA, the authors introduce a penalty term to the error function in order to guarantee the diversity of ensemble members. The performance of the proposed algorithm is evaluated by applying it to train a class of feedforward neural networks to solve the large n-bit parity problems. By comparing with the previous studies, the experimental results revealed that the neural network ensemble classifiers trained by the ESTSA have very good classification ability.

Keywords: Artificial neural networks, orthogonal array, ensemble, n-bit parity problems.

1 Introduction

Artificial neural networks (ANNs) have been applied to many application areas and have gained remarkable success. The back-propagation (BP) algorithm is the well-known training algorithm for feedforward neural networks. The drawback of the BP is that it may be trapped in local optima and its activation function must be differentiable. Thus, many evolutionary algorithms had been proposed to train the connection weights and/or the architectures of neural networks. Yao [1] gave an elaborate survey in 1999. Recently Mendes *et al.* [2] proposed a particle swarm optimization algorithm for feedforward neural network training. Nikolaev and Iba [3] hybridized the genetic programming and the backpropagation algorithm to train the polynomial feedforward

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 389–395, 2010.

neural networks. Tasi *et al.* proposed a hybridized algorithm, which combines the Taguchi method and the genetic algorithm, to tune the structure and parameters of a neural network [4]. Recently, Tseng and Chen proposed the TPGLS [5], the two-phase genetic local search algorithm, to train feedforward neural networks, and the authors observed that it is important to keep a good balance between the global search and the local search.

For complicated problems, to train the ANNs is not easy. Instead of using a single ANN, combining multiple simpler neural networks is adopted to tackle the complex problems. On combining ANNs, there are two main approaches: ensemble-based and modular [6]. In this paper, the proposed training algorithm is based on the first approach. As mentioned in [6], the ensemble has better generalization performance than any individual network for classification problems. But, it is not easy to construct the ANN ensemble. While constructing the ANN ensemble, there are two main issues: the selection of a set of ANNs to be the members of the ensemble and the combination of the outputs of the members of the ensemble. Sharkey pointed out that the members of the ensemble must generalize differently; otherwise there is no advantage to combining them together. Thus, the key of success is to keep the diversity of members of the ensemble as well as the accuracy [7]. In ensemble, if any two members make different errors on any instance, they are diverse. As usual, we want the degree of diversity to be as higher as possible among ensemble members. Sharkey presented an overview of the main methods for creating ensemble members [6]. The readers are further referred to [8] for the valuable survey of diversity creation methods. In [9], Liu and Yao proposed a learning algorithm, called the negative correlation learning (NCL), to train ANN ensembles. In the NCL, a correlation penalty term was added into the error function in order to produce negative correlated members. Based on the NCL, Liu *et al.* [10] proposed evolutionary ensembles with the NCL (EENCL) in which fitness sharing and the NCL are used to maintain the diversity of different members of the population. Recently, Yao and Islam [11] gave a review of research on evolutionary approaches for constructing ANN ensembles.

For the non-linearly separable property, the n-bit parity problem is known as a difficult classification problem. Especially when n is large, it can hardly be solved in reasonable time, except the special designed methods [12][13]. Thus, it is usually used as a benchmark for testing new ANN training algorithms.

In our recent research [14], we presented the systematic trajectory search algorithm (STSA) to train feedforward neural networks. The STSA was tested on training feedforward neural networks to solve the n-bit parity problems of various sizes and two real medical diagnosis problems. The experimental results show that the feedforward neural networks trained by the STSA have very good classification ability. But, for large n-bit parity problems, the STSA spent a lot of time to train the ANNs. In this paper, based on the STSA, the authors propose an evolutionary method for constructing ANN ensembles to solve some larger n-bit parity problems. The experimental results revealed the effectiveness of the proposed method.

The remainder of the paper is organized as follows. Section 2 introduces the proposed method for constructing the ANN ensemble. Section 3 gives the experiments and results, and finally Section 4 concludes the paper.

2 The Evolutionary ANN Ensemble

The flowchart of the STSA is shown in Fig. 1 (with shaded blocks removed). In the global search phase, the orthogonal array (OA) is utilized to generate uniformly distributed initial solutions. Among these solutions, *k* best ones are selected as seeds. Then in the local search phase, a local search method called random array local search (RALS) is applied to each seed. The details of the STSA can be found in [14]. Based on the STSA, we propose an evolutionary method, called ESTSA, for constructing ANN ensembles to solve classification problems. As mentioned above, there are two main issues for constructing ensembles: the selection of a set of ANNs to be the members of the ensemble and the combination of the outputs of the members of the ensemble. The detail of the constructing method of ANN ensembles is described in Fig. 1.

2.1 Method for Creating Ensemble Members

In this paper, we incrementally construct the ensemble from the evolutionary population in each epoch. Inspired by the NCL [9][10], we introduce a penalty term into the error function in order to maintain the higher diversity among the members of the ensemble during evolving the population. For the simplicity of validation, without loss of generality, we focus on the two-class classification problems where the output of the ANN classifier is 1 or 0. The error function and the penalty term of the *i*th member of the population are defined as follows:

$$E_i = \frac{1}{N}\sum_{j=1}^{N}\left|f_{ij} - d_j\right| + \alpha\frac{1}{NK}\sum_{j=1}^{N}\left(\sum_{k=1}^{K} p_{ijl_k}\right)^{\beta}, \tag{1}$$

$$p_{ijl_k} = \left|f_{ij} - d_j\right| * \left|f_{l_k j} - d_j\right|, \tag{2}$$

where N is the number of instances, f_{ij} and d_j are the *i*th member's output and the desired output for the *j*th instance, respectively, and K is the size of the ensemble, p_{ijl_k} is the penalty term, α and β are user-defined parameters to adjust the weight of the penalty term. In the equation (2), $f_{l_k j}$ is the output of the *k*th ensemble member for the *j*th instance. If the *i*th member of the population misclassifys the *j*th instance as well as the *k*th ensemble member does, p_{ijl_k} will be 1. Otherwise, p_{ijl_k} will be 0. Thus, the penalty term will enforce each individual in the population to avoid misclassifying the same instances which are misclassified by the ensemble members and thus guarantee the diversity of ensemble members.

Updating Rules of Ensemble. In this paper, the maximum size of ensemble is fixed during constructing period. The following are the rules for updating the ensemble.

Step1. If the size of the ensemble does not exceed the upper bound, add the new member into the ensemble and then return.

Step2. Otherwise, replace the worst member of the ensemble with the new member if the new member is better than the worst member of the ensemble.

2.2 Method of Combining the Outputs

There are many kinds of combining methods for producing the output of the ensemble. This study just focused on the classification problems, thus the majority voting was adopted.

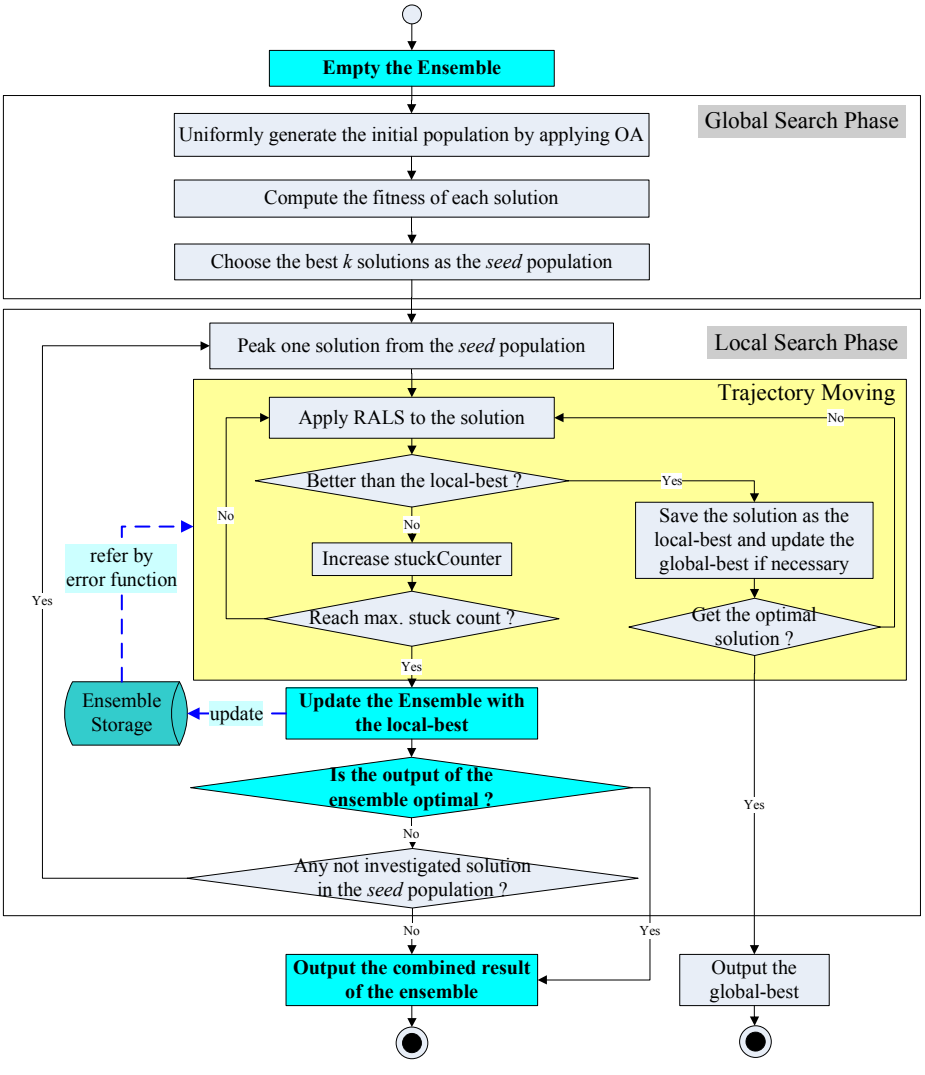


Fig. 1. The flowchart of the ESTSA for constructing ANN Ensemble

3 Experiments and Results

The *n*-bit parity problem, which is not linearly separable, is one of the most difficult classification problems. So we used the *n*-bit parity problems to test the classification

ability of the feedforward neural networks trained by the proposed algorithm. The program was coded in Java and run on a personal computer with Intel Core 2 Quad/Q6600 2.40Ghz CPU and 1024 MB memory. The java program was implemented with single threading mode that means only one CPU serves each running program even though the computer has an Intel Core 2 Quad CPU.

3.1 The Settings of Experiments

In order to compare the results with those of the previous study [14], ten runs were conducted for each of the neural network architectures: *N*-1-1-1-1 with *N*=8, 9, 10, 11 and 12. For *N*=13, 14 and 15, five, three and two runs were conducted respectively. After doing some preliminary study, we decided that the hidden nodes used the sign activation function which will output either -1 or 1, and the output nodes used the step activation function which will output 0 or 1. All the 2^n instances were used in training phase. When *n* is 15, the number of instances is 32768 which is very large for any classifier. The parameters used in the proposed algorithm are shown in Table 1. The number of iteration of the RALS varies for different *n* in the STSA. As can be seen, for larger *n*, the number of iteration of the RALS is larger in order to search the neighborhood more thoroughly in the STSA. On the other hand, the number of iteration of the RALS is the same with different *n* in the ESTSA for saving the searching time. But, in order to explore the neighborhood more thoroughly, the search involves more dimensions and it visits more neighbors.

Table 1. Parameters used in the STSA and the ESTSA for the n-bit parity problem

Parameters		Value in STSA	Value in ESTSA
Size of the ***seed*** population (*k*)		30	30
Orthogonal Array		OA(243, 20, 3, 3)	OA(243, 20, 3, 3)
RALS	Number of neighbors in the neighborhood	300	500
	Levels of each dimension	10	10
	Number of iterations of RALS	100 for *N*= 8, 9 200 for *N*=10, 11	10
	minDim	1	1
	maxDim	3	#dimension/4
	vRatio	0.3	0.3
Maximum stuck counter		5	20
Size of Ensemble (maximum)		N/A	5
Parameters of the Penalty (α, β)		N/A	(1.0, 1.0)

3.2 Experimental Results

The experimental results are listed in Table 2 for the *n*-bit parity problems. In Table 2, the value of each entry stands for the average computation time over some experimental runs.

For some larger n-bit parity problems, the STSA spent a lot of time to train the networks with architecture N-2-2-2-1. For example, when *N* is 11, the STSA needed more than nine hours to train an ANN classifier. Besides, STSA still failed to gain an optimal solution in some cases. The results in Table 2 also show that the ESTSA is more effective and more efficient than the STSA while applying them to solve the large *n*-bit parity problems.

Table 2. Comparison between the ESTSA and the STSA. The results of the ESTSA were averaged over the complete runs. (*1) The value is the average of nine successful runs out of ten complete experiments in the STSA; (*2) The value is the average of five successful runs out of ten complete experiments in the STSA.

	STSA [14] (N-2-2-2-1)	**ESTSA (N-1-1-1-1)**	
	Average CPU time	**Average CPU time**	**Average size of ensemble**
N=8	0 hour 8 min.	1 min. 27 sec. (10/10 runs)	3.2
N=9	0 hour 25 min.	4 min. 59 sec. (10/10 runs)	4.4
N=10	3 hour 33 min. (*1)	15 min. 25 sec. (10/10 runs)	4.0
N=11	9 hour 37 min. (*2)	46 min. 42 sec. (10/10 runs)	5.0
N=12	N/A	1 hour 28 min. (9/10 runs)	4.6
N=13	N/A	6 hour 26 min. (3/5 runs)	5.0
N=14	N/A	13 hour 28 min. (3/3 runs)	5.0
N=15	N/A	26 hour 17 min. (1/2 runs)	5.0

4 Conclusions

In the previous study, the authors presented the systematic trajectory search algorithm (STSA) to train the ANN and gained some valuable results. The STSA utilizes the orthogonal array (OA) to uniformly generate the initial population in order to globally explore the solution space, and then applies a novel trajectory search method to exploit the promising areas thoroughly. In this paper, an evolutionary constructing algorithm, called the ESTSA, of the ANN ensemble is proposed. Based on the STSA, we introduced a penalty term into the error function in order to guarantee the diversity of ensemble members and incrementally construct the ensemble by applying the ESTSA to search the solution space. The ESTSA was applied to solve some large n-bit parity problems. When *n* is 15, the number of instances is 32768 which is very large for general classifiers. The experimental results reveal that the ESTSA is more effective and more efficient than the STSA. It can train a feedforward neural network with a simple architecture (N-1-1-1-1) to classify 15-bit parity problem.

For future studies, we plan to apply the ESTSA to classify some medical datasets of the UCI machine learning repository. We also plan to do some research works on the updating mechanism of the ensemble.

References

[1] Yao, X.: Evolving Artificial Neural Networks. Proc. IEEE 87(9), 1423–1447 (1999)
[2] Mendes, R., Cortez, P., Rocha, M., Neves, J.: Particle Swarms for Feedforward Neural Network Training. In: Proceedings of the International Joint Conference on Neural Networks, pp. 1895–1899. IEEE Press, New York (2002)
[3] Nikolaev, N., Iba, H.: Learning Polynomial Feedforward Neural Networks by Genetic Programming and Backpropagation. IEEE Trans. Neural Netw. 14(2), 337–350 (2003)
[4] Tsai, J.T., Chou, J.H., Liu, T.K.: Tuning the Structure and Parameters of a Neural Network by Using Hybrid Taguchi-Genetic Algorithm. IEEE Trans. Neural Netw. 17(1), 69–80 (2006)
[5] Tseng, L.Y., Chen, W.C.: A Two-Phase Genetic Local Search Algorithm for Feedforward Neural Network Training. In: Proceedings of the International Joint Conference on Neural Networks, pp. 5221–5225. IEEE Press, New York (2006)
[6] Sharkey, A.J.C.: On Combining Artificial Neural Nets. Connect. Sci. 8(3/4), 299–314 (1996)
[7] Hansen, L.K., Salamon, P.: Neural Network Ensembles. IEEE Trans. Pattern Anal. Mach. Intell. 12(10), 993–1001 (1990)
[8] Brown, G., Wyatt, J., Harris, R., Yao, X.: Diversity Creation Methods: A Survey and Categorisation. Journal of Information Fusion 6, 5–20 (2005)
[9] Liu, Y., Yao, X.: Ensemble Learning via Negative Correlation. Neural Networks 12, 1399–1404 (1999)
[10] Liu, Y., Yao, X., Higuchi, T.: Evolutionary Ensembles with Negative Correlation Learning. IEEE Trans. on Evol. Comput. 4(4), 380–387 (2000)
[11] Yao, X., Islam, M.M.: Evolving Artificial Neural Network Ensembles. IEEE Computational Intelligence Magazine 3(1), 31–42 (2008)
[12] Stork, D.G., Allen, J.D.: How to Solve the N-bit Parity Problem with Two Hidden Units. Neural Networks 5(6), 923–926 (1992)
[13] Hohil, M.E., Liu, D., Smith, S.H.: Solving the N-bit Parity Problem Using Neural Networks. Neural Networks 12(9), 1321–1323 (1999)
[14] Tseng, L.Y., Chen, W.C.: The Systematic Trajectory Search Algorithm for Feedforward Neural Network Training. In: Proceedings of International Joint Conference on Neural Networks, pp. 1174–1179. IEEE Press, New York (2007)

Multiattribute Bayesian Preference Elicitation with Pairwise Comparison Queries

Shengbo Guo[1] and Scott Sanner[2]

[1] ANU/NICTA, Locked Bag 8001, Canberra ACT 2601, Australia
[2] NICTA/ANU, Locked Bag 8001, Canberra ACT 2601, Australia
{Shengbo.Guo,Scott.Sanner}@nicta.com.au
http://users.rsise.anu.edu.au/~sguo/

Abstract. Preference elicitation (PE) is an very important component of interactive decision support systems that aim to make optimal recommendations to users by actively querying their preferences. In this paper, we present three principles important for PE in real-world problems: (1) multiattribute, (2) low cognitive load, and (3) robust to noise. In light of three requirements, we introduce an approximate PE framework based on a variant of TrueSkill for performing efficient closed-form Bayesian updates and query selection for a multiattribute utility belief state — a novel PE approach that naturally facilitates the efficient evaluation of value of information (VOI) for use in query selection strategies. Our VOI query strategy satisfies all three principles and performs on par with the most accurate algorithms on experiments with a synthetic data set.

Keywords: preference elicitation, decision-making under uncertainty.

1 Introduction

Preference elicitation (PE) is an important component of eCommerce and recommender systems that propose items or services from a potentially large set of available choices but due to practical constraints may only query a limited number of preferences. The PE task consists of (a) querying the user about their preferences and (b) recommending an item that maximizes the user's latent utility. Of course, a PE system is limited by real-world performance constraints that require phase (a) to be efficient while ensuring phase (b) can make an optimal recommendation with high certainty. To this end, we outline five principles important for the practical application of PE in real-world settings used to guide our research in this work:

1. *Multiattribute*: Exploiting the natural attribute structure of services or items in the form of multiattribute utility functions [10] is crucial when the number of recommendable items exceeds the number of queries a PE system can reasonably ask. In this case, learning preferences over attribute dimensions can simultaneously inform preferences over many items.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 396–403, 2010.

2. *Low cognitive load*: Since the task of utility elicitation is cognitively difficult and error prone [5], queries that are more difficult for users lead to higher noise and less certainty in the utility elicited. Thus, we focus on pairwise comparison queries known to require low cognitive load for users [6].
3. *Robust to noise*: A real-world PE system has to make robust utility predictions in the presence of noisy query responses. Bayesian PE approaches that maintain a belief distribution over utility functions and update beliefs using a realistic query confusion model are one natural way to handle noise, although exact inference in these Bayesian models may often be intractable.

In the following sections, we develop an approximate Bayesian PE framework to satisfy all three of these principles and demonstrate this empirically on a synthetic dataset.

2 Bayesian Preference Elicitation

2.1 User Utility Model

In multiattribute utility theory (MAUT) [10], utilities are modeled over a D-dimensional *attribute set* $\mathcal{X} = \{X_1, \ldots, X_D\}$ with *attribute choices* $X_d = \{x_{d1}, \ldots, x_{d|X_d|}\}$ (where $|X_d|$ denotes the cardinality of X_d). An *item* is described by its attribute choice assignments $\mathbf{x} = (x_1, \ldots, x_D)$ where $x_d \in X_d$. In our model, an *attribute weight vector* $\mathbf{w} = (w_{11}, \ldots, w_{1|X_1|}, \ldots, w_{D1}, \ldots, w_{D|X_D|})$ describes the utility of *each* attribute choice in *each* attribute dimension.

We assume that the utility $u(\mathbf{x}|\mathbf{w})$ of item $\mathbf{x}$ w.r.t. attribute weight vector $\mathbf{w}$ decomposes *additively* over the attribute choices of $\mathbf{x}$, i.e.,

$$u(\mathbf{x}|\mathbf{w}) = \sum_{d=1}^{D} \mathbf{w}_{d,\#(\mathbf{x},d)}, \quad u^*(\mathbf{x}) = \sum_{d=1}^{D} \mathbf{w}^*_{d,\#(\mathbf{x},d)} \tag{1}$$

where $\#(\mathbf{x}, d)$ returns index in $\{1, \ldots, |X_d|\}$ for attribute choice x_d of $\mathbf{x}$ and u^* represents the user's true utility w.r.t. their true (but hidden) $\mathbf{w}^*$.

Since $\mathbf{w}^*$ is unknown to the decision support system, it is the goal of preference elicitation to learn an estimate $\mathbf{w}$ of $\mathbf{w}^*$ with enough certainty to yield a low expected loss on the item recommended. We take a Bayesian perspective on learning $\mathbf{w}$ [3] and thus maintain a probability distribution $P(\mathbf{w})$ representing our beliefs over $\mathbf{w}^*$.

Because $P(\mathbf{w})$ is a distribution over a multidimensional continuous random variable $\mathbf{w}$, we represent this distribution as a Gaussian with diagonal covariance, represented compactly in a factorized format as follows:

$$P(\mathbf{w}) = \prod_{d=1}^{D} \prod_{i=1}^{|X_d|} p(w_{di}) = \prod_{d=1}^{D} \prod_{i=1}^{|X_d|} \mathcal{N}(w_{di}; \mu_{di}, \sigma^2_{di}). \tag{2}$$

We assume the vectors μ and σ represent the respective mean and standard deviation for the normal distribution over each corresponding attribute choice in $\mathbf{w}$.

2.2 PE Graphical Model and Inference

In this paper, we take a Bayesian approach to PE. Thus, given a prior utility belief $P(\mathbf{w}|R^n)$ w.r.t. a (possibly empty) set of $n \geq 0$ query responses $R^n = \{q_{kl}\}$ and a new query response q_{ij}, we perform the following Bayesian update to obtain a posterior belief $P(\mathbf{w}|R^{n+1})$ where $R^{n+1} = R^n \cup \{q_{ij}\}$:

$$\begin{aligned} P(\mathbf{w}|R^{n+1}) &\propto P(q_{ij}|\mathbf{w}, R^n)P(\mathbf{w}|R^n) \\ &\propto P(q_{ij}|\mathbf{w})P(\mathbf{w}|R^n) \end{aligned} \tag{3}$$

Assuming that our query likelihood $P(q_{ij}|\mathbf{w})$ is modeled as an indicator function, we note that the form of the exact posterior is not a diagonal Gaussian as is the initial prior $P(\mathbf{w}|R^0) = P(\mathbf{w}|\emptyset) = P(\mathbf{w})$ defined in (2), rather, it is a mixture of truncated Gaussians where the number of mixture components grows exponentially with the number of queries.

To avoid this exponential exact inference, we must turn to approximate Bayesian inference techniques. First we note that the use of (3) leads to a slight variation on the *TrueSkill*$^{\text{TM}}$ [8] graphical model for multiattribute PE shown in Figure 1.

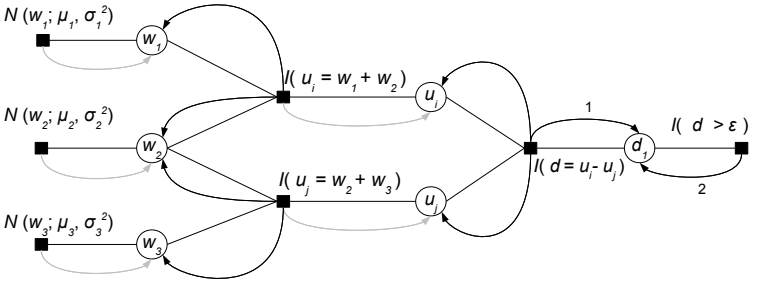


Fig. 1. PE factor graph variant of TrueSkill for $q_{ij} = i \succ j$. Items i and j have two attribute choices each with respective weights (w_1, w_2) and (w_2, w_3) (note that i and j share the common attribute choice with weight w_2). The posterior over (w_1, w_2, w_3) can be inferred with the following message passing schedule: (1) messages pass along *gray* arrows from left to right, (2) the marginal over d is updated via message 1 followed by message 2 (which required moment matching), (3) messages pass from right to left along *black* arrows.

Of key importance in this approximate Bayesian updating scheme is to note that from prior sufficient statistics μ^n and σ^n for $P(\mathbf{w})$ in the form of (2), the update with the $n+1$st query response q_{ij} results in posterior sufficient statistics μ^{n+1} and σ^{n+1}. While not guaranteed in practice due to approximation, ideally we would expect that in the limit of queries as $n \to \infty$, our belief distribution will approach full certainty in the user's hidden utility, i.e., $\mu^n \to \mathbf{w}^*$ and $\sigma^n \to \mathbf{0}$.

3 Value of Information

Now that we know how to efficiently update our multiattribute utility distribution based on a user's query responses, we are left with the question of how to formulate a query strategy. While all queries should improve the certainty of our utility estimate w.r.t. some items, we are most concerned with finding the optimal item with high certainty.

One way to evaluate different queries is to measure the extent to which they help the PE system reach this optimal decision, which can be formalized using *value of information* (VOI) [9]. VOI plays an important role in many Bayesian PE strategies, as first proposed in [5] and our Bayesian PE framework naturally facilitates an approximation of VOI as we show next.

One way to formalize the VOI of a query in our PE framework is to note that the query which maximizes our VOI is the one that most reduces our loss. Unfortunately, we can never know our *true loss* for recommending an item, we can only calculate our *expected loss* — the query leading to the maximum reduction in expected loss will then maximize our expected VOI.

But how do we define the expected loss at any stage of PE? First we note that if we stop PE after eliciting query response set R, then we have posterior utility beliefs $P(\mathbf{w}|R)$ summarized by sufficient statistics (μ^R, σ^R). From this, we can efficiently compute the highest expected utility item i_R^*[1]:

$$i_R^* = \arg\max_i \ \mathrm{E}_{P(\mathbf{w}|R)}[u(i|\mathbf{w})] = \arg\max_i \ u(i|\mu^R). \tag{4}$$

This straightforward result exploits the fact that $P(\mathbf{w}|R)$ is diagonal Gaussian and thus the expectation factorizes along each attribute dimension.

Now let us assume that we have access to the true utilities of items i and k, respectively $u^*(i)$ and $u^*(k)$ recalling (1). If we recommend item i in place of item k, then our loss for doing so is $\max(0, u^*(k) - u^*(i))$, i.e., if $u^*(k) > u^*(i)$ then we lose $u^*(k) - u^*(i)$ by recommending i, otherwise we incur no loss.

Of course, we do not have the true item utilities to compute the actual loss. However, in the Bayesian setting, we do have a belief distribution over the item utilities, which we can use to compute the expected loss. Thus, to compute the expected loss (EL) of recommending item the best item i_R^* instead of recommending item k, we evaluate the following expectation:

$$\mathrm{EL}(k, R) = \mathrm{E}_{P(\mathbf{w}|R)}\left[\max\left(0, u(k|\mathbf{w}) - u(i_R^*|\mathbf{w})\right)\right] \tag{5}$$

Unfortunately, the computation of EL is difficult because the expectation integral over the max prevents the calculation from factorizing along attribute dimensions of the Gaussian utility beliefs. For this reason, we opt for a computationally simpler approximation of the expected loss ($\widehat{\mathrm{EL}}$) where we use the expected utility $u(i_R^*|\mu^R)$ of i_R^* from (4) as a surrogate for its true utility, leading to the *closed-form* calculation:

[1] We assume any item is synonomous with its feature vector (e.g., i_R^* and $\mathbf{x}_{i_R^*}$ are used interchangeably).

$$\widehat{\mathrm{EL}}(k,R) = (\mu_{i_R^*} - \mu_k)(1 - \Phi_{\mu_k,\sigma_k^2}(\mu_{i_R^*})) - \frac{\sigma_k}{\sqrt{2\pi}} \exp\left(-\frac{(\mu_{i_R^*} - \mu_k)^2}{2\sigma_k^2}\right) \quad (6)$$

Here, Φ_{μ_k,σ_k^2} is the normal CDF, $\mu_k = \sum_d \mu_{d,\#(k,d)}$, $\sigma_k^2 = \sum_d \sigma^2_{d,\#(k,d)}$, and $\mu_{i_R^*} = \sum_d \mu_{d,\#(i_R^*,d)}$. (Space limitations require omission of the derivation.)

From this single item expected loss, we can determine the maximum expected loss (MEL) we might incur by recommending i_R^* instead of *some* other k:

$$\mathrm{MEL}(R) = \max_k \widehat{\mathrm{EL}}(k,R). \quad (7)$$

From MEL, we can finally approximate the expected reduction in loss — the expected VOI (EVOI) — of obtaining query response q_{ij} for items i and j:

$$\mathrm{EVOI}(R,i,j) = -\mathrm{MEL}(R) + \sum_{q_{ij}} \left[\mathrm{E}_{P(\mathbf{w}|R)} P(q_{ij}|\mathbf{w})\right] \mathrm{MEL}(R \cup \{q_{ij}\}) \quad (8)$$

4 PE Query Selection Strategies

A query strategy specifies what comparison query between item i and item j should be asked when given the current query response set $R^n = \{q_{kl}\}$ after n queries have been asked.

Ideally, we are supposed to propose the query q_{ij} such that the user's response can maximally reduce the expected loss with regard to the belief on $\mathbf{w}$, i.e.,

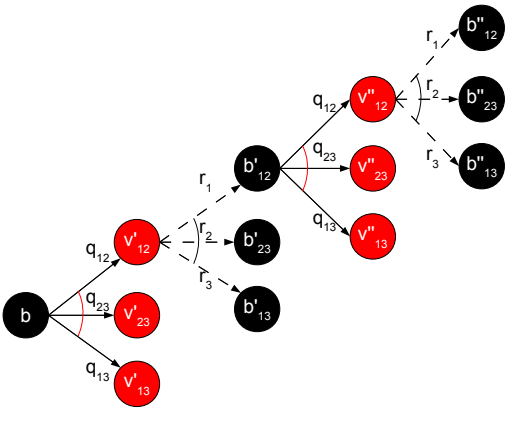


Fig. 2. Illustration of two-step look ahead with maximization and expectation query strategy (incomplete for illustration purpose). For a given belief state b, suppose there are three possible queries, a_1, a_2, a_3, each with three possible responses, we represent the intermediate belief state of asking a_1 at b by $v_1^{'}$. For given $v_1^{'}$, we use $b_{11}^{'}$ to represent the posterior belief when observing response r_1. Same operation applies to $b_{11}^{'}$ and yields the two-step look ahead search tree. The operators associates with red circle and black circle represent *expectation* and *maximization*, respectively.

the expected value of information. We adopt a two-step look ahead strategy involving maximization and expectation to find the best query (Figure 2), and we call it PE-ME. Given the current belief state and each possible query, we simulate the user's response and evaluate the posterior belief state. Then, we apply the same operation to each posterior belief state. Once we compute the EVOI for each query and possible user's response, we compute the expectation of EVOI of asking a query with different user responses, and select the query that maximizes the EVOI for each query. Then, we repeat the expectation and maximization operations, and select the query with maximal EVOI.

Aside from the above strategies, we also experimented with the PE query strategies that do not use VOI heuristics. The first query strategy is *Random Two*, a baseline strategy that randomly picks two items for a query and serves as an upper bound for worst-case performance. The second is *Preference elicitation upper confidence bounds (PE-UCB)*, which queries the items with the largest and second largest upper confidence bounds. Given μ (mean) and σ (standard deviation) of the item belief, the upper confidence bounds are $\mu + c\sigma$, where $c > 0$ is a constant. We use $c = 1$ (PE-UCB(1)), $c = 2$ (PE-UCB(2)), and $c = 3$ (PE-UCB(3)), respectively.

5 Experimental Results

5.1 Data Set and User Simulation

We evaluate our approach using a synthetic data set. For this data set, we generate items with all combinations of three item attributes of interest, with 2, 2, and 5 choices, respectively, making 20 items total. In this dataset, we assume all attribute combinations are feasible.

To simulate the user response process, we drew random utilities for the attribute choice vector $\mathbf{w}$ according to two models: (a) a uniform distribution over $[1, 100]$ for each attribute choice, and (b) a normal distribution with mean μ drawn uniformly from $[1, 100]$ for each attribute choice, and sampling random positive semidefinite matrices for use as full covariance matrices.

5.2 Results

All of the following experiments were implemented in Matlab (code available on request), under Windows, using an Intel(R) Core$^{\text{TM}}$2 Quad CPU Q9550, 2.83GHz, 3Gb RAM PC. $\epsilon = 5$ for Bayesian updates.

We show a plot of the normalized average loss $(\max_j u^*(j) - u^*(i_R^*))$ of all algorithms vs. the number of query responses elicited in Figure 3. That is, on the y-axis, we show for 50 averaged trials what fraction of the total loss was incurred by each algorithm after the x-axis specified number of queries. A result of 0 indicates no loss and is optimal.

For uniformly distributed utility distributions, 'PE-ME' query strategy outperforms 'Random Two' and 'PE-UCB(c)' when less than 4 queries were asked.

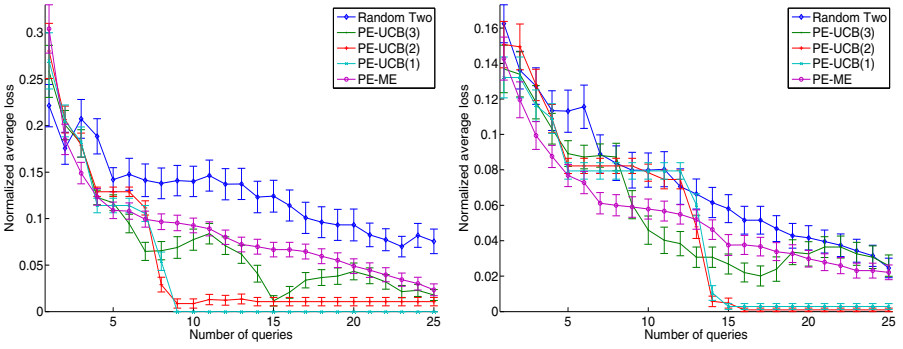


Fig. 3. Normalized average loss vs. number of queries for various PE strategies. Error bars indicate standard error.

After that, all the 'PE-UCB(c)' strategies start to get better performance. Amongst those strategies, the performance of 'PE-UCB(3)' is the best when less than 8 queries are asked, however, it is not stable. Since the ninth queries, 'PE-UCB(2)' and 'PE-UCB(1)' become the best two. For Gaussian utility functions, 'PE-ME' achieves the best performance till ninth query is asked. It turns out that 'PE-ME' is very effective in reducing normalized average when a few queries are asked.

6 Related Work

Space limitations prevent a thorough literature review; we briefly discuss how related work addresses the three PE principles, which is summarized Section in 1. While various early PE research influenced many of the design decisions in this work [5, 4, 1] such as the Bayesian modeling approach, factorized belief representation, and VOI, these papers typically relied on either *standard gamble* queries requiring users to state their preference over a probability distribution of outcomes or they directly elicit utility values. While theoretically sound, these

Table 1. Comparison among PE algorithms in terms of three requirements

Literature	Multiattribute	Low cognitive load	Robustness
[5]	√		√
[4]	√		√
[1]			√
[7]		√	√
[6]		√	
[12]	√		
Our approach	√	√	√

methods may require high cognitive load for elicitation, and thus are prone to error [5]; we rely on pairwise comparison queries known to require low cognitive load [6].

7 Conclusion

In light of the PE requirements, we developed an effective Bayesian PE framework based on a variant of TrueSkill for performing efficient closed-form multiattribute utility belief updates — a novel PE approach that facilitated efficient closed-form VOI approximations for PE query selection. This contrasted with related work that failed to satisfy all requirements. As demonstrated on the synthetic data set, the 'PE-ME' query strategy is multiattribute, low cognitive load via pairwise queries, robust to noise.

References

[1] Boutilier, C.: A POMDP formulation of preference elicitation problems. In: AAAI, pp. 239–246 (2002)
[2] Bradley, R.A., Terry, M.E.: Rank analysis of incomplete block designs: The method of paired comparison. Biometrika 39, 324–345 (1952)
[3] Chajewska, U., Koller, D.: Utilities as random variables: Density estimation and structure discovery. In: UAI, pp. 63–71 (2000)
[4] Chajewska, U., Koller, D., Ormoneit, D.: Learning an agent's utility function by observing behavior. In: ICML, pp. 35–42 (2001)
[5] Chajewska, U., Koller, D., Parr, R.: Making rational decisions using adaptive utility elicitation. In: AAAI, pp. 363–369 (2000)
[6] Conitzer, V.: Eliciting single-peaked preferences using comparison queries. Journal of Artificial Intelligence Research 35, 161–191 (2009)
[7] Doshi, F., Roy, N.: The permutable POMDP: fast solutions to POMDPs for preference elicitation. In: AAMAS, vol. 1, pp. 493–500 (2008)
[8] Herbrich, R., Minka, T., Graepel, T.: Trueskill^{TM}: A Bayesian skill rating system. In: NIPS, pp. 569–576 (2006)
[9] Howard, R.A.: Information value theory. IEEE Transactions on Systems Science and Cybernetics 2(1), 22–26 (1966)
[10] Keeney, R.L., Raiffa, H.: Decisions with Multiple Objectives: Preferences and Value Tradeoffs. John Wiley & Sons, Chichester (1976)
[11] Kschischang, F.R., Frey, B.J., Loeliger, H.-A.: Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 47(2), 498–519 (2001)
[12] Viappiani, P., Boutilier, C.: Regret-based optimal recommendation sets in conversational recommender systems. In: RecSys (2009)

Local Bayesian Based Rejection Method for HSC Ensemble

Qing He[1], Wenjuan Luo[1,2], Fuzhen Zhuang[1,2], and Zhongzhi Shi[1]

[1] The Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100080, China
[2] Graduate University of Chinese Academy of Sciences, Beijing 100049, China
{heq,luowj,zhuangfz,shizz}@ics.ict.ac.cn

Abstract. Based on Jordan Curve Theorem, a universal classification method, called Hyper Surface Classifier (HSC) was proposed in 2002. Experiments showed the efficiency and effectiveness of this algorithm. Afterwards, an ensemble manner for HSC(HSC Ensemble), which generates sub classifiers with every 3 dimensions of data, has been proposed to deal with high dimensional datasets. However, as a kind of covering algorithm, HSC Ensemble also suffers from rejection which is a common problem in covering algorithms. In this paper, we propose a local bayesian based rejection method(LBBR) to deal with the rejection problem in HSC Ensemble. Experimental results show that this method can significantly reduce the rejection rate of HSC Ensemble as well as enlarge the coverage of HSC. As a result, even for datasets of high rejection rate more than 80%, this method can still achieve good performance.

Keywords: HyperSurface Classification (HSC); HSC Ensemble; Rejection.

1 Introduction

Among various kinds of classification algorithms in machine learning, there exists a family of covering algorithms (also called rule learning algorithms) which follow the so-called separate-and-conquer or covering strategy. In detail, these methods learn a rule at a time, which explains(covers) a part of the training examples, then the covered examples are removed and the rest examples are used to learn new rules successively [1]. This procedure is carried on until all the training examples are covered. While in the classifying phase, for a test example, rules are tried until it satisfies any of the learned rules and then it is classified as the label implied by the satisfied rule.

Different covering algorithms differ in the way how single rules are generated. Based on the McCulloch-Pitts Neural Model, Zhang et al. [2] proposed a covering algorithm for classification which models an M-P neuron as a covering on the input space and constructs a set of labeled sphere neighborhoods for classification. Based on the same model, Wu et al. [3] combined the kernel function algorithm of SVM and spherical domain covering algorithm of constructive machine learning method, and made improvements for the algorithm.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 404–412, 2010.

Based on Jordan Curve Theorem, He et al. [4] proposed another covering algorithm for classification which constructs hypersurface homeomorphic to lower dimensional sphere of the input space, and class labels are obtained according to whether the intersecting number between the test sample and the hypersurface is odd or even. In fact, He et al. [4] view the hypersurface as single rules and generate them recursively, while Zhang et al. [2] and Wu et al. [3] consider the labeled sphere neighborhoods to be single rules and construct them iteratively.

In machine learning and data mining, a typical assumption is that the training data and the test data are drawn from the same feature space and follow the same distribution [5]. However, in the real world, distributions of training data and test data do not actually match, and then rejection happens.

There are mainly two types of rejection, one rises when a new class label appears only in the test data [6], and the other rises typically in covering algorithms when test samples belong to the areas which no classifier(or covering) covers [3]. In case of the former kind of rejection, there are several strategies to handle, such as *distance-based reject-option* [7], *ambiguity reject-option* [7] and *combinations of one-class and supervised classifiers* [8].

However, the classification threshold or the rejection threshold for the above mentioned methods are difficult to choose [6].

In case of the latter kind of rejection, Zhang et al. [9] put forward a probabilistic model which utilizes a Gaussian kernel covering function, they also implement Expectation Maximization Algorithm for global optimization. As a result, this model broadens the application domain of their covering algorithm and reduces the rejection rate. However, it remains a difficult problem on how to select a proper kernel function.

As a kind of covering algorithm also suffering from rejection, HSC was proposed to classify spirals in [4]. In case of real world data, He et al. [10] proposed a dimension reduction method for HSC which transforms high dimensional dataset into three-dimensional dataset. Afterwards, for high dimensional classification, Zhao et al. [11] proposed HSC Ensemble, which divides a dataset vertically into sub datasets with every three dimensions of data, and then ensembles sub classifiers generated by every sub dataset. However, Zhao et al. [11] did not take rejection into consideration. In this paper, we study the rejection problem of HSC Ensemble and only discuss the second type of rejection, i.e., only rejection when test examples exceed the boundary of HSC is investigated. The rest of this paper is organized as follows: in Section 2, we outline the main idea of hyper surface classifier (HSC). Then in Section 3, we focus on HSC Ensemble and the rejection method, the results of which are presented in Section 4, and Section 5 concludes our paper.

2 Main Idea of HSC Algorithm

HSC is a universal classification method based on Jordan Curve Theorem in topology. Compared to SVM, this approach can directly solve the nonlinear classification problem in the original space other than higher dimensional space, therefore without the use of kernel function.

Jordan Curve Theorem. Let X be a closed set in n-dimensional space . If X is homeomorphic to a sphere in n-dimensional space, then its complement has two connected components, one called inside, the other called outside. Based on the Jordan Curve Theorem, n-dimensional space can be separated by a double-sided surface that is homeomorphic to a sphere in $(n-1)$-dimensional sphere. So X can be seen as a separating hyper surface.

Classification Theorem. For any given point $x \in R^n/X$, x is inside of X, iff, the intersecting number between any radial from x and X is odd; and x is outside of X, iff, the above-mentioned intersecting number is even.

Based on the above theorems, we can construct hypersurface and use it for classification. Main steps are in the following.

Main Steps of HSC. There are two major procedures in HSC, one is the training procedure, and the other is the testing procedure.

Training Procedure

Step 1. Input the training samples, containing k categories and d-dimensions. Let the training samples be distributed within a rectangular region.

Step 2. Divide the region into $\overbrace{10 \times 10 \times \cdots \times 10}(10^d)$ small regions called units. Here, 10 is the number of divisions we perform on each dimension practically, and actually it could be any number that is above 1.

Step 3. If there are some units containing samples from two or more different categories, then divide them into smaller units recursive until each unit covers at most samples from the same category.

Step 4. Label each unit with $1, 2, \cdots, k$ according to the category of the samples inside, and unite the adjacent units with the same labels into a bigger unit.

Step 5. For each unit, save its contour as a link, and this represents a piece of hyper surface.

Testing Procedure

In the testing procedure, we just count the number of intersections between the radial starting from the test sample and the trained hypersurface. If the number is odd, then the sample is marked as the class indicated by the link.

The classification algorithm based on hypersurface is a polynomial algorithm if the same class samples are distributed in finite connected components. Experiments showed that this method can work fairly well in both accuracy and efficiency in three-dimensional space even for large size data up to 10^7. Specifically, [12] pointed out that models trained from Minimal Consistent Subset can correctly classify all the remaining points in the sample set.

3 HSC Ensemble and Rejection Method

3.1 HSC Ensemble

By attaching equal importance to each feature, HSC Ensemble firstly groups the overall dataset into sub datasets with every three dimension of data, when there

is less than three dimension for the last sub dataset, one dimension or two from the last but one sub dataset is reused. Detailed algorithms can be found in [11].

3.2 Rejection Problem

Fig. 1 illustrates the rejection problem in two dimensional space, where the hypersurface(also called HSC Classifier or HSC model in this paper) is constructed from combinations of rectangles. Suppose there are two classes in the figure, one is covered by the hypersurface on the bottom left corner (depicted with "+"), and the other is covered by the hypersurface on the top right corner (depicted with "."). When test data are located in the covering rectangles, there is no rejection. However, when test data come from the ellipse region, which is not covered by any of the hypersurface, all the data points in this region become unrecognized and rejection happens.

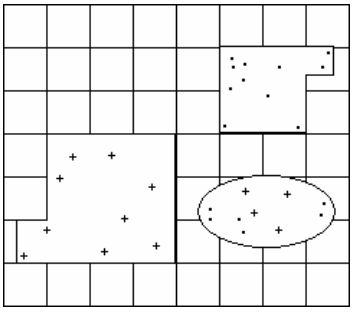


Fig. 1. Rejection Problem in 2 Dimension

3.3 Local Bayesian Based Rejection Method

HSC divides the feature space recursively until all pure covers(hypersurface) are constructed. The way that HSC covers feature space is so conservative that a cover only covers points exactly from the same class. Moreover, from the main steps of HSC, we can see that, after each dividing, the edge length of HSC decreases to 1/10 of the original edge length before dividing.

As a result, the more deeply HSC divides the feature space, the shorter the edge of HSC would be and the more unrecognized fragments there would be in the feature space. In an extreme case, each HSC cover may cover only 1 point with extremely short edges. Thus when class labels are assigned to points in a haphazard manner, the feature space has to be divided recursively in order to get pure HSC covers. Consequently, small fractions of attribute range are rejected. In case of high dimensionality, the coverage of HSC may shrink exponentially when too many fractions of attribute range are rejected.

From the above analysis, there are two strict constraints restricting the coverage of HSC. The first is that each piece of hypersurface should only cover points

that have determinate class labels the same as the hypersurface. The second is that edge length of hypersurface becomes smaller during recursive dividing.

Encouraged by the above analysis, we proposed the following method to deal with rejection problem in HSC. From the bayesian inference theory we have:

$$P(C|X) = \frac{P(C,X)}{P(X)} = \frac{P(X|C) \times P(C)}{P(X)} \tag{1}$$

In Equation (1), X denotes a sample, $P(X)$ is the probability of X, while C is a class label and $P(C)$ corresponds to its probability. $P(C|X)$ denotes the probability of class label C conditioned on X and $P(X|C)$ is the probability of X conditioned on C. For a classification problem, the goal is to find the class label C which has the highest $P(C|X)$. As $P(X)$ is the same for any class label, so the classification problem is to estimate the probability $P(X|C) \times P(C)$, as to find the class label C with the maximal $P(C,X)$.

However, when dealing with rejection, it becomes unrealistic to estimate $P(C,X)$ as there is no hypersurface covering X. In order to solve this problem, we choose points from the local area of X to estimate the probability $P(C,X)$. Moreover, instead of viewing X from 3-dimensional perspective as from HSC, we analyze X from each-dimensional perspective and then ensemble the probability $P(C,X)$ for all possible C.

Here, suppose the dimension number of X is d, and X is represented as $X(x_1, x_2, x_3, ..., x_d)$, we define the local area of X to be:

$$Local\_Area(X) = \{P(p_1, ..., p_d) | P \in TS, \exists i \in (1, ..., d), (\lfloor x_i \rfloor \leq p_i < \lceil x_i \rceil)\} \tag{2}$$

In Equation (2), $P(p_1, ..., p_d)$ is a point from training set (denoted as TS), while $\lfloor x_i \rfloor$ denotes the largest integer smaller than x_i and $\lceil x_i \rceil$ is the smallest integer larger than x_i. From the equation above, we can see that for a point which has at least one dimension with attribute value p_i falling into the scope of $\lfloor x_i \rfloor$ and $\lceil x_i \rceil$, it will be included in $Local\_Area(X)$. This is different from the cover of hypersurface which follows:

$$Cover(H) = \{P(p_1, ..., p_d) | P \in TS, \forall i \in (1, ..., d), (h_{i_{low}} \leq p_i < h_{i_{up}})\} \tag{3}$$

In Equation (3), P is the same as in Equation (2), while $H([h_{1_{low}}, h_{1_{up}}), [h_{2_{low}}, h_{2_{up}}), ..., [h_{d_{low}}, h_{d_{up}}))$ stands for a piece of hypersurface, where $[h_{i_{low}}, h_{i_{up}})$ composes the edge of H in the i_{th} dimension. Compare Equation (2) with Equation (3), we can see that the local area has a much more smooth boundary, while an HSC Classifier has a so strict boundary that many points would be rejected. Moreover, the edge length of local area remains 1, however, based on the dividing mechanism of HSC, the edge length of HSC is ≤ 1 ("=" happens only in the first dividing of feature space). In consequence, $Local\_Area(X)$ has a much larger coverage than $Cover(H)$. Take a 2-dimensional example, a point $A(2.1, 2.2)$ corresponding to a hypersurface of $H : \{X(x,y) | 2 \leq x < 3, 2 \leq y < 3\}$ and a local area $L : \{X(x,y) | 2 \leq x < 3\} \bigcup \{X(x,y) | 2 \leq y < 3\}$, another point $B(2.9, 3.1)$ would not belong to the hypersurface H but fall into L.

Since we only aim to maximize $P(C,X)$ in the $Local\_Area(X)$, therefore, when we have the $Local\_Area(X)$ constructed, calculating the probability of

$P(C, X)$ in $Local\_Area(X)$ is equal to calculating the probability of $P(C)$ in $Local\_Area(X)$, as all X are from the same $Local\_Area(X)$. For a test example X that is rejected, detailed operations for classification are as follows:

Step 1. Scan the training data and construct $Local\_Area(X)$ defined in Equation (2).
Step 2. For each class label C, for all the points in $Local\_Area(X)$, compute $P(C)$.
Step 3. Select the class label C which has the highest $P(C)$ and predict X as label C.

In step 2, we compute $P(C)$ in the $Local\_Area(X)$, as mentioned above, maximizing this $P(C)$ is equivalent to maximizing the $P(C, X)$ in $Local\_Area(X)$. And we call the above algorithm as local bayesian based rejection method(LBBR). On the time complexity of the above algorithm, suppose N is the number of training instances, d is the number of dimensions and c is the number of total class labels. The average-run-time of Step 1 is $O(d * N)$. Since we use a hash tree to store class labels, the average-run-time of Step 2 is $O(\alpha * d * N)$, where α is the percentage of points in the $Local\_Area$ to the whole dataset, so $\alpha < 1$. For Step 3, the average-run-time is $O(c)$. As c is much smaller than $d * N$, thus the overall average time is $O(d * N + \alpha * d * N) = O(\beta * d * N)$, with $\beta < 2$.

Therefore, the time complexity of local bayesian based rejection(LBBR) method is linear with the size of training data. Thus LBBR has a good scalability. Experimental results are given in Section 4. Also, we make use of the ROC (Receiver Operating Characteristic) space for a clear comparison between HSC with LBBR (denoted as "HSC+LBBR" in the following) and HSC without.

4 Experiments

4.1 Experimental Data

We use data sets from UCI repository [13] to evaluate our rejection method. Table 1 gives the detailed description of data sets adopted. We divide training sets and testing sets in the same way as [11], i.e., we randomly choose about 2/3 of data for training and the rest for testing. For some of the datasets in Table 1, we delete some ineffective or redundant attributes. Take the dataset image for example, we delete 2 attributes which are not important for classification but actually impede the construction of sub HSC Classifiers.

Table 1. Description of data Sets

Data Set	Number of Dimensions	Number Of training Samples	Number Of testing Samples	Number Of classes
hayes	5	80	71	3
breast	9	369	200	2
glass	9	151	54	7
image	17	150	60	7
parkins	21	130	65	2
inosphere	34	250	101	2
libras	90	300	59	15

4.2 Experimental Results

We implemented LBBR and results are shown in Table 2. In Table 2, Recall denotes the accuracy of HSC on training data and HSC+LBBR stands for the accuracy of "HSC+LBBR" on test data. Accuracy of HSC denotes, for all the test samples that are **not** rejected, the accuracy of HSC. And Accuracy of LBBR is, for all the rejected test samples, the accuracy of LBBR. From Rejection Rate column, we can see that different datasets have different rejection rates. Besides, we can see from the table that when rejection rate is high, the overall accuracy mainly depends on the accuracy of LBBR. Thus when local Bayesian performs well, such as on the dataset libras, the overall accuracy would be high.

Table 2. Results of HSC Ensemble on Rejection Problem

Data Set	Recall	Rejection Rate	Accuracy of HSC	Accuracy of LBBR	HSC+LBBR
hayes	86.25%	50.7%	88.57%	72.22%	80.28%
breast	99.46%	24.85%	99.99%	96.34%	99.09%
glass	100.00%	98.14%	100.00%	66.04%	66.67%
image	100.00%	100.00%	-	85.00%	85.00%
parkins	100.00%	100.00%	-	76.92%	76.92%
inosphere	100.00%	95.05%	99.95%	91.67%	92.08%
libras	100.00%	100.00%	-	91.53%	91.53%

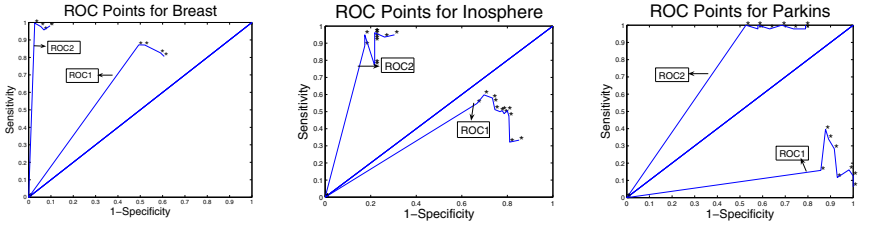


Fig. 2. ROC **Points** of HSC Classifiers With LBBR And Without

Fig. 2 shows the ROC **points** of HSC Classifiers on 3 different datasets, corresponding to Breast, Inosphere and Parkins, respectively. In the ROC space, each point is coordinated as $(1 - specificity, sensitivity)$. For specificity and sensitivity, we have:

$$Specificity = \frac{TN}{TN + FP}, Sensitivity = \frac{TP}{TP + FN} \tag{4}$$

In the above equations, TP denotes the number of True Positives, FN-False Negatives, TN-True Negatives and FP-False Positives. Thus a ROC point describes the classification capability of a classifier. For a binary classification problem, an ideal classifier would obtain a sensitivity of 1 and a specificity of 1, thus, in the ROC space, it would be represented as the point(0,1). Classifiers that take random guess with $Sensitivity = 1 - Specificity$, would be lined as $y = x$. Thus,

classifiers better than random guess should be represented as points above the line($y = x$) in the ROC space.

In all subgraphs of Fig. 2, ROC2 includes the ROC **points** of HSC+LBBR classifiers, and ROC1 includes the ROC **points** of HSC classifiers. All points on ROC2 are closer to (0,1) than points on ROC1, which indicates that sub HSC+LBBR classifiers outperform sub HSC classifiers. The larger is the area between ROC2 and ROC1, the more effective is LBBR. Looking at the rejection rate of breast, inosphere and parkins in Table 2, we can see that the area between ROC1 and ROC2 becomes larger as the rejection rate grows. From the discussion and comparisons above, we can see that LBBR for HSC is effective.

5 Conclusion

In this paper, after analyzing the cause of rejection in HSC Ensemble, we propose a local bayesian based rejection method(LBBR) to solve the rejection problem in HSC Ensemble. This method computes the probability of a test sample belonging to a certain class in the local area of the test sample and get the class label of test sample in use of Bayesian method. As a result, this method not only smooths the edge of hypersurface, but also enlarges the coverage of HSC Ensemble. As shown by our experiments, this method obtains high accuracy as well as obvious improvement for HSC classifiers. In conclusion, the local bayesian based method could deal with the rejection problem of HSC Ensemble effectively.

Acknowledgements

This work is supported by the National Science Foundation of China (No.60675010, 60933004, 60975039), 863 National High-Tech Program (No.2007AA01Z132), National Basic Research Priorities Programme (No.2007CB311004) and National Science and Technology Support Plan (No.2006BAC08B06).

References

1. Furnkranz, J.: ROC n Rule Learning Towards a Better Understanding of Covering Algorithms. Machine Learning 58, 39–77 (2005)
2. Zhang, L., Zhang, B.: A Geometrical Representation of McCullochCPitts Neural Model and Its Applications. IEEE Transactions on Neural Networks 10(4) (1999)
3. Wu, T., Zhang, L., Yan-Ping, Z.: Kernel Covering Algorithm for Machine Learning. Chinese Journal of Computers 28(8) (2005)
4. He, Q., Shi, Z.-Z., Ren, L.-A., Lee, E.S.: A Novel Classification Method Based on Hyper Surface. International Journal of Mathematical and Computer Modeling, 395–407 (2003)
5. Pan, S.J., Yang, Q.: A Survey on Transfer Learning. IEEE Transactions on Knowledge and Data Engineering (October 12, 2009)
6. Landgrebe, T.C.W., Tax, D.M.J., Paclk, P., Duin, R.P.W.: The interaction between classification and reject performance for distance-based reject-option classifiers. Pattern Recognition Letters 27(8), 908–917 (2006)

7. Dubuisson, B., Masson, M.: A statistical decision rule with incomplete knowledge about classes. Pattern Recognition 26(1), 155–165 (1993)
8. Landgrebe, T., Tax, D., Paclk, P., Duin, R., Andrew, C.: A combining strategy for ill-defined problems. In: Fifteenth Ann. Sympos. of the Pattern Recognition Association of South Africa, pp. 57–62 (2004)
9. Zhang, L., Wu, T., Zhou, Y., Zhang, Y.P.: Probabilistic Model for Covering Algorithm. Journal of Software 18(11), 2691–2699 (2007)
10. He, Q., Zhao, X., Shi, Z.: Classification based on dimension transposition for high dimension dataInternational Journal Soft Computing-A Fusion of Foundations. Methodologies and Applications, 329–334 (2006)
11. Zhao, X.R., He, Q., Shi, Z.Z.: HyperSurface Classifiers Ensemble for High Dimensional Data sets. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 1299–1304. Springer, Heidelberg (2006)
12. He, Q., Zhao, X.-R., Shi, Z.-Z.: Minimal consistent subset for hyper surface classification method. International Journal of Pattern Recognition and Artificial Iintelligence 22(1) (2008)
13. http://archive.ics.uci.edu/ml/

Orthogonal Least Squares Based on Singular Value Decomposition for Spare Basis Selection

Min Han and De-cai Li

School of Electronic and Information Engineering, Dalian University of Technology, Dalian 116023, China
minhan@dlut.edu.cn

Abstract. This paper proposes an improved orthogonal least square algorithm based on Singular Value Decomposition for spare basis selection of the linear-in-the-weights regression models. The improved algorithm is based on the idea of reducing meaningless calculation of the selection process through the improvement of orthogonal least square by using the Singular Value Decomposition. This is achieved by dividing the original candidate bases into several parts to avoid comparing among poor candidate regressors. The computation is further simplified by utilizing the Singular Value Decomposition to each sub-block and replacing every sub-candidate bases with the obtained left singular matrix, which is a unitary matrix with lower dimension. It can avoid the computation burden of the repeated orthogonalisation process before each optimal regressor is determined. This algorithm is applied to the linear-in-the-weights regression models with the predicted residual sums of squares (PRESS) statistic and minimizes it in an incremental manner. For several real and benchmark examples, the present results indicate that the proposed algorithm can relieve the load of the heave calculation and achieve a spare model with good performance.

Keywords: singular value decomposition, orthogonal least square, predicted residual sums of squares (PRESS) statistic, spare bases selection.

1 Introduction

In many situations, multivariate time series are required to model the complex dynamic of chaotic systems [1, 2]. It has been shown that predictions using multivariate time series may be significantly better than those using univariate time series [3]. Although the multivariate inputs can provide more information for modeling, the increment of the inputs also means more complex model structure, which would produce very poor generalization and heavy time consuming. In order to achieve accurate predictions with multivariate inputs, the effective complexity of the model has to be controlled based on the principle that ensures the smallest possible model that fits the training data well.

Several methods have been developed for simplifying the model complexity, which are typically divided into constructive approaches [4, 5], and pruning methods [6, 7]. In the constructive approach, the structure of the network is incrementally built through adding nodes to the hidden layer one by one or group by group, while the pruning

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 413–421, 2010.

approach starts with an initial selection of a large number of hidden units which is reduced as the algorithm proceeds.

In a recent publication, Billings [8] introduced the forward orthogonal least squares (OLS) algorithm for model construction. For a large class of linear-in-the-weights model, the orthogonal least square method has been known as an useful method for model complexity control and the ill-conditioning problems can also be solved effectively. To achieve a spare model with directly optimizing model generalization capability, a full automated procedure is proposed by using the PRESS statistic as a cost function for iterative model evaluation [9]. In [10], the OLS and a D-optimality criterion are used to determine the structure as for optimizing the model approximation ability, spare, and robust simultaneously. However, In the OLS algorithm, to select a candidate regressor, the vectors formed by the candidate regressors must be processed by using orthogonal methods, which is time consuming.

In the present paper, an orthogonal least squares based on Singular Value Decomposition for Spare Basis Selection is proposed (OLS-SVD). The new algorithm divides the candidate regressors into some sub-blocks to avoid comparing among the poor neurons and uses the forward orthogonal least squares algorithm based on the SVD approach to select the candidate regressors. The paper is organized as follows. Section 2 briefly reviews some primarily acknowledge on Linear-in-the-weights regression model. The OLS-SVD algorithm based on SVD and Press statistic is described in section 3. In section 4, two examples are simulated to illustrate the performance of the new algorithm. Finally, the conclusions of this paper are given in section 5.

2 Linear-in-the-Weights Regression Model

Consider a discrete nonlinear dynamical system of the form

$$y(k) = f\left(y(k-1),\ldots,y\left(k-n_y\right),u(k-1),\ldots,u\left(k-n_u\right)\right)+e(k) \qquad (1)$$

where $f(\cdot)$ is an unknown nonlinear mapping, $u(k)$ and $y(k)$ are the input and output variables of the system at discrete time step k, n_u and n_y represent the maximal orders in $u(k)$ and $y(k)$, respectively, while e(k) is assumed to be Gaussian noise with zero mean and unit variance. A *linear-in-weights* regression model of the form (2) can approximate $f(\cdot)$ with zero error.

$$\begin{aligned} y(k) &= \hat{y}(k)+e(k)=\sum_{i=1}^{n_M}\theta_i\Phi_i\left(x(k)\right)+e(k) \\ &= \boldsymbol{\Phi}^T(k)\boldsymbol{\theta}+e(k) \end{aligned},\ k=1,\ldots,N \qquad (2)$$

where N is the size of the observation data set, $\hat{y}(k)$ is the model output, θ_i are the model weights, $\boldsymbol{\theta} = [\theta_1,..., \theta_{nM}]^T$, $\Phi_i(x(k))$ are the regressors and $\boldsymbol{\Phi}(k) = [\Phi_1(x(k)),..., \Phi_{nM}(x(k))]^T$, $x(k)=[y(k-1),\ldots,y(k-n_y),u(k-1),...,u(k-n_u)]^T$ denotes the system input vector, and n_M is the total number candidate regressors.

The above N equations can be written compactly as

$$\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\theta}+\mathbf{e} \qquad (3)$$

where $\mathbf{\Phi} = [\boldsymbol{\Phi}(1),..., \boldsymbol{\Phi}(N)]^T$, where $\boldsymbol{\Phi}_i = [\Phi_i(1),..., \Phi_i(N)]^T$, $1 \le i \le n_M$, and defining $\mathbf{y} = [y(1),..., y(N)]^T$, $\mathbf{e} = [e(1),..., e(N)]^T$.

The forward selection algorithm is often used to construct a parsimonious model with a subset of $n_0 \ll n_M$ regressors by some model-selective criterion, among which the leave-one-out cross validation are metrics that measures a model's generalization capability.

Let $\{x(k), y(k)\}_{k=1:N,-k}$, be the resulting data set by removing the kth data point from the training data set $\{x(k), y(k)\}_{k=1:N}$, and denote estimated model output with j regressors as $\hat{y}_{j,-k}(k)$, and the related predicted residual at k as $\varepsilon_{j,-k}(k)$. For a linear-in-the-weights model with n_M candidate regressors, the PRESS errors are calculates as

$$\begin{aligned} \varepsilon_{n_M,-k}(k) &= y(k) - \hat{y}_{n_M,-k}(k) \\ &= \frac{\varepsilon_{n_M}(k)}{1 - \Phi^T(k)(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\Phi(k)} \end{aligned} \tag{4}$$

where $\varepsilon_{n_M}(k) = y(k) - \hat{y}_{n_M}(k)$

3 OLS_SVD Algorithm Based on SVD and Press Statistic

It appears that the computation burden of choosing the best subset model, which minimizes the mean square PRESS error $E[\varepsilon_{n_M,-k}(k)]$, will be expensive, as the matrix inversion involving. However, if employing an orthogonal forward regression to incrementally minimize PRESS error as presented in [11, 12], the model selection procedure would become computationally affordable.

Consider the linear-in-the-weights regression model (3), several OLS algorithms have been developed for selecting the candidate regressiors, such as Classical Gram-Schmidt (CGS) algorithm, Modified Gram-Schmidt (MGS) algorithm and Householder algorithm. The three methods have the same drawbacks to select candidate regressors with a forward selection procedure, as almost all the unjustified regressors need an orthogonal process with the predetermined ones before each optimal column is selected, and the computational complexity of the orthogonalisation procedure increases with the number of columns has been selected. Moreover, at each selection step, the PRESS statistic or other selective criterion with each unjustified column has to be formed for comparison. It appears that the computation burden increases as the growing of model size, and repeated comparing among the poor candidate regressors would also make the computation effort of the selection procedure extensive.

To overcome these drawbacks, an improved orthogonal least square algorithm based on SVD is proposed for spare model construction. The new algorithm divides the design matrix into several sub-blocks to avoid comparing among poor candidate regressors. And then SVD is applied to each sub-block, which would avoid the repeated orthogonalisation process by replacing the sub-design matrix with the obtained orthogonal singular matrix. Based on the model form of (3), the algorithm is showed as follows.

Firstly, dividing the design matrix $\mathbf{\Phi}_{all}$ into d parts equally by column, $\mathbf{\Phi}$ is a sub-block derived from $\mathbf{\Phi}_{all}$ with n_M (n_M<n_{Mall}) columns. Assuming the rank of $\mathbf{\Phi}$ is p in columns, and $\mathbf{\Phi}$ can be decomposed according to the Singular Vector Decomposition theorem as

$$\mathbf{\Phi} = \mathbf{U}\begin{bmatrix} \mathbf{\Sigma} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix}\mathbf{V}^H \tag{5}$$

where $\mathbf{\Sigma} = diag\left(\sigma_1, \sigma_2, \cdots, \sigma_p\right)$, and the diagonal elements of Σ are in the order $\sigma_1 \geq \sigma_2 \geq, \ldots, \geq \sigma_p$, $\mathbf{U}$ is an $N \times N$ orthogonal matrix consisting of all the orthogonalized eigenvectors associate with the eigenvalues of $\mathbf{\Phi}\mathbf{\Phi}^T$, and $\mathbf{V}$ is and $n_M \times n_M$ orthogonal matrix. Then only the first r eigenvectors (associated with the larger eigenvalues) are retained, while (p-r) smaller components are discarded, assuming that the latter describe mostly noise, and the r selected eigenvalues satisfy

$$\frac{\sum_{i=1}^{r}\sigma_i^2}{\sum_{j=1}^{p}\sigma_j^2} > \eta_0 \tag{6}$$

where η_0 is a user defined parameter, $0 < \eta_0 < 1$. According to the r eigenvalues, blocking the matrix $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{\Sigma}$ as

$$\mathbf{V} = [\mathbf{V}_1 \mid \mathbf{V}_2], \quad \mathbf{U} = [\mathbf{U}_1 \mid \mathbf{U}_2], \quad \mathbf{\Sigma} = [\mathbf{\Sigma}_1 \mid \mathbf{\Sigma}_2] \tag{7}$$

where $\mathbf{V}_1$ is a n_M by r Matrix, $\mathbf{V}_2$ is a M by (n_M-r) Matrix, $\mathbf{U}_1$ is a N by r Matrix, $\mathbf{U}_2$ is a N by (N-r) Matrix, and $\mathbf{\Sigma}$ is a r by r Matrix, $\mathbf{V}_2$ is a (p-r) by (p-r) Matrix.

Therefore, by neglecting the small singular values, which can be shown that mainly represent noise, Eq. (5) can be simplified by $\mathbf{\Phi}_r$, whose rank equals the number of remaining singular values.

$$\mathbf{\Phi}_r = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^H \tag{8}$$

where $\mathbf{U}_1 = \left[\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_r\right]$.

Then, based on the approximation matrix $\mathbf{\Phi}_r$, the linear-in-the-weights regression model (3) can be expressed as

$$\mathbf{Y} = \mathbf{\Phi}\boldsymbol{\theta} + \mathbf{e} \approx \mathbf{\Phi}_r\boldsymbol{\theta} + \mathbf{e} = \mathbf{U}_1 \cdot \mathbf{\Sigma}_1\mathbf{V}_1^H \cdot \boldsymbol{\theta} + \mathbf{e} \tag{9}$$

Define $\mathbf{g} = \mathbf{\Sigma}\mathbf{V}_1^H \cdot \boldsymbol{\theta} = \left[g_1, g_2, \cdots, g_M\right]^T$, and Eq. (9) can be rewritten as

$$\mathbf{Y} = \mathbf{U}_1\mathbf{g} + \mathbf{e} \tag{10}$$

Eq. (8) is similar to the linear-in-the-weights regression model (3), but there are several significant differences between the two candidate regression matrix, $\mathbf{U}_1$ and $\mathbf{\Phi}$. First, compared to the matrix $\mathbf{\Phi}$, $\mathbf{U}_1$ is an orthogonal matrix, which means all the candidate regressors are already orthogonal with each other, and the repeated orthogonalisation decomposition of $\mathbf{\Phi}$ is avoided in the forward selection process. Second, by neglecting the small singular values in the matrix $\mathbf{\Sigma}$, $\mathbf{U}_1$ is an N by r orthogonal matrix consisting of only r orthogonalized eigenvectors associate with the r most significant eigenvalues of $\mathbf{\Phi}\mathbf{\Phi}^T$. Substituting $\mathbf{\Phi}$ with $\mathbf{U}_1$ would reduce the number of the candidate regressors,

and avoid the comparing among the poor regressors at each selection step, which is time consuming and complicated.

Then, the PRESS statistic can be embodied as a selective criterion for model construction, and the formulation is similar to the case given in [9]. For the improved orthogonal model (8) with r candidate regressors, the PRESS errors are calculates as

$$\begin{aligned}\varepsilon_{n,-k}(k) &= y(k) - \hat{y}_{n,-k}(k) \\ &= \frac{\varepsilon_n(k)}{1 - \mathbf{U}_1^T(k)(\mathbf{U}_1^T\mathbf{U}_1)^{-1}\mathbf{U}_1(k)} = \frac{\varepsilon_n(k)}{\eta_n(k)}\end{aligned} \tag{11}$$

where $\mathbf{U}_1$ is an orthogonal matrix.

To measure the generalization capability of a sub-model with n candidate regressors, the mean square PRESS error is the given by averaging all these PRESS errors.

$$J_n = E[\varepsilon_{n,-k}^2(k)] = E\left[\left(\frac{\varepsilon_n(k)}{\eta_n(k)}\right)^2\right] = \frac{1}{N}\frac{\varepsilon_n^2(k)}{\eta_n^2(k)} \tag{12}$$

Note that the model residual $\varepsilon_n(k)$ for the n-term model can be computed recursively as

$$\varepsilon_n(k) = y(k) - \sum_{i=1}^{n} u_i(k) g_i = \varepsilon_{n-1}(k) - u_n(k) g_n \tag{13}$$

where

$$g_i = \frac{\mathbf{u}_i^T \mathbf{y}}{\mathbf{u}_i^T \mathbf{u}_i} \tag{14}$$

And similarly, the PRESS error weighting $\eta_n(k)$ can be written in a recursive formula by

$$\eta_n(k) = 1 - \sum_{i=1}^{n} \frac{u_i^2(k)}{\mathbf{u}_i^T \mathbf{u}_i} = \eta_{n-1}(k) - \frac{u_n^2(k)}{\mathbf{u}_n^T \mathbf{u}_n} \tag{15}$$

Assume $\mathbf{\Phi}_1, \mathbf{\Phi}_2, \ldots, \mathbf{\Phi}_d$ are the d sub-blocks of the design matrix $\mathbf{\Phi}_{all}$ and $\tilde{\mathbf{U}}_i$ (i=1,2,…,d) is the ith subsets obtained from the selection process described above with each $\mathbf{\Phi}_i$, then the final basis function for model (3) is reconstructed as $\tilde{\mathbf{U}} = [\tilde{\mathbf{U}}_1, \tilde{\mathbf{U}}_2, ..., \tilde{\mathbf{U}}_d]$. If the size of $\tilde{\mathbf{U}}$ is still a large number, we can repeat this strategy till an appropriate model structure is achieved. It is worth to notice that d is a user defined parameter. The computation of selection procedure is simpler as the increase of d, for the candidate regressors in each sub-blocks is small. However, the computation burden of SVD needed is also increase as more subsets have to be considered. Given no a priori knowledge, there is no automated optimal way of choosing d, so in a practical scenario, we depend on the user's experiment and familiarity with size of data available to guide the choice of this parameter.

4 Simulations

The sparse modeling procedure described in the previous sections is applied to the simulated and benchmark data set, respectively.

4.1 Approximation of SinC Function with Noise

In this example, a RBF network with the Gaussian basis function is employed to approximate the SinC function, which is a popular choice to illustrate Support Vector Machine for regression (SVR) in the literature

$$y(x) = \begin{cases} \dfrac{\sin(x)}{x} & x \neq 0 \\ 1 & x = 0 \end{cases} \tag{16}$$

The Gaussian basis function employed is given by

$$\Phi_i(\mathbf{x}) = \exp\left(-\frac{\| \mathbf{x} - \mathbf{c}_i \|^2}{2\sigma^2}\right) \tag{17}$$

and $\mathbf{c}_i$ and σ are the centers and the widths of the basis functions, respectively.

The proposed algorithm is firstly compared with an OLS method based on and PRESS statistic to illustrate its efficiency in computation cost, in which the orthogonalisation process is achieved by employing a modified Gram-Schmidt (MGS) method. Moreover, for comparing the influence of the size of training set, 5 experiments are provided for each algorithm. In these experiments, the training set $\{x(k), y(k)\}$ is created from $y(x)$ where the input $x(k)$ is uniformly distribution on the interval (-10, 10). The number of the training sample is given in Table 1, which changes from 200 to 1000. In order to make the regression problem real, the Gaussian noise with zero mean and standard deviation 0.2 has been added to all the training samples. A testing set $\{x(k), y(k)\}$ is also created from $y(x)$ with two hundred noise free data, in which $x(k)$ is equally spaced in (-10, 10). The optimal kernel width is found to be σ^2=10 empirically, and the candidate center is taken as each training data point $x(k)$. Therefore, the number of regressors in each experiment is the same as that of the training set $\{x(k), y(k)\}$.

Table 1. Performance comparison of computation cost and accuracy for the simulated data set

Algorithm	Training time	Training MSE	Testing MSE	Training size	Number of sub-blocks
OLS with PRESS	2.9312	0.0429	0.0424	200	1
OLS-SVD *with* PRESS	0.2320	0.0408	0.0421	200	1
OLS with PRESS	12.0031	0.0287	0.0294	400	2
OLS-SVD *with* PRESS	0.8125	0.0290	0.0292	400	2
OLS with PRESS	26.5750	0.0331	0.0330	600	3
OLS-SVD *with* PRESS	2.1125	0.0305	0.0310	600	3
OLS with PRESS	47.7125	0.0235	0.0244	800	4
OLS-SVD *with* PRESS	3.9688	0.0237	0.0238	800	4
OLS with PRESS	73.5094	0.0172	0.0179	1000	5
OLS-SVD *with* PRESS	6.8024	0.0179	0.0185	1000	5

Table 1 compares the performance of the two OLS based method in the term of the computation cost and the modeling accuracy. As observed from Table 1, both the algorithms are comparable in accuracy. However, in the term of computation time, the OLS algorithm based on PRESS statistic is more time consuming for the repeated

orthogonalisation process in each selection step, and the training time increases dramatically with the number of the sample size. For the proposed algorithm, the model selection procedure is computation efficient by dividing the original design matrix into sub blocks and employing the SVD to simplify the orthogonalisation process. Compared with the OLS algorithm based on PRESS statistic, training time of the proposed algorithm increases slowly with the sample size.

4.2 Approximation of Nonlinear Dynamic Control System

Consider the following nonlinear dynamic control system

$$z(k)=\frac{z(k-1)z(k-2)z(k-3)u(k-2)(z(k-3)-1)+u(k-1)}{1+z^2(k-2)+z^2(k-3)} \tag{18}$$

A training set $\{x(k), y(k)\}$ and testing set $\{x(k), y(k)\}$ are created form (18) with 200 and 100 samples, respectively, where the system input u(k) is uniformly distribution on the interval [-1, 1]. In order to make the problem real, the Gaussian noise with zero mean and standard deviation 0.05 has been added to all the training and testing samples.

A RBF network with the thin-plate-spline basis function is employed to follow the dynamic process

$$\Phi_i\left(\mathbf{x}(k)\right)=\left\|\mathbf{x}(k)-\mathbf{c}_i\right\|^2\log\left(\left\|\mathbf{x}(k)-\mathbf{c}_i\right\|\right) \tag{19}$$

And the system input vector is denoted as

$$\mathbf{x}(k)=\left[\,y(k-1)\,y(k-2)\,y(k-3)u(k-1)u(k-2)\right]^T \tag{20}$$

As each training data point $\mathbf{x}(k)$ is considered as a candidate center of the network, there are 200 candidate regressors. In addition, for the small sample size, no dividing process is provided in this example.

To illustrate the generalization ability of the proposed algorithm, it is also compared with 5 other model construction methods, such as the LROLS algorithm with PRESS statistic, the OLS algorithm with PRESS statistic, the LROLS algorithm with MSE, the RVM algorithm, and the enhanced k-means clustering and least squares (CLS). All these algorithms have been employed in [9].

Table 2. Performance comparison of model size and accuracy for the nonlinear system

Algorithm	Validation set used	Model size	Training MSE	PRESS statistic	Testing MSE
OLS with PRESS [9]	No	51	0.002280	0.003864	0.005187
LROLS with PRESS [9]]	No	31	0.003192	0.003706	0.005892
LROLS with MSE[[9]	No	42	0.001883	0.003067	0.004872
RVM [9]	No	42	0.001598	0.002577	0.004935
CLS [9]	Yes	49	0.003940	0.007607	0.005580
OLS-SVD with PRESS	No	30	0.002527	0.003517	0.005210

Table 2 illustrate that, the six algorithms are comparable with each other in the term of model size and accuracy. In this example, the 200 candidate regressors are reduced to a size of 66 first, and a sparse model with 30 terms is achieved after the forward selection process. The model size of the proposed algorithm is comparable with the LROLS algorithm with PRESS statistic, while much smaller than the other four algorithms. However, although the LROLS with PRESS can construct a parsimonious model, its modeling accuracy is the worse, compared with the other five algorithms. For the proposed algorithm, as only the small singular values, which can be shown that mainly represent noise, are neglected in the process of SVD, the reconstructed candidate regressors in model (10) kept most of the useful information. Hence, the proposed algorithm can be acceptable in modeling accuracy. In addition, as same as the OLS with PRESS, the LROLS with PRESS and the RVM, the proposed algorithm is also an automated procedure without any requirement of a validation set.

5 Conclusions

In this paper, a new OLS algorithm based on SVD for linear-in-the-weights regression models is proposed. This is achieved by dividing the original candidate bases into several parts to avoid comparing among poor candidate regressors. The computation is further simplified by utilizing the Singular Value Decomposition to each sub-block. It can avoid the computation burden of the repeated orthogonalisation process before each optimal regressor is determined and further reduce the number of the candidate regressors. The results obtained from the examples which include the SinC function and a nonlinear dynamic control system demonstrate its effectiveness and accuracy.

Acknowledgements

This research is supported by the project (60674073) of the National Nature Science Foundation of China and the project (2007AA04Z158) of the National High Technology Research and Development Program of China (863 Program).

References

[1] Chen, S.M., Hwang, J.R.: Temperature prediction using fuzzy time series. IEEE Transactions on Systems, Man and Cybernetics -Part B 30(2), 263–275 (2000)
[2] Coulibaly, P., Anctil, F., Bobee, B.: Multivariate reservoir inflow forecasting using temporal neural networks. Journal of Hydrologic Engineering 6(5), 367–376 (2001)
[3] Han, M., Wang, Y.J.: Analysis and modeling of multivariate chaotic time series based on neural network. Expert System with Applications 36(2), 1280–1290 (2009)
[4] Feng, G., Huang, G.B., Lin, Q., Gay, R.: Error Minimized Extreme Learning Machine with Growth of Hidden Nodes and Incremental Learning. IEEE Transactions on Neural Networks 20(8), 1352–1357 (2009)
[5] Huang, G.B., Chen, L., Siew, C.K.: Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes. IEEE Transactions on Neural Networks 17(4), 879–892 (2006)

[6] Salmerón, M., Ortega, J., Puntonet, C.G., Prieto, A.: Improved RAN sequential prediction using orthogonal techniques. Neurocomputing 41, 153–172 (2001)
[7] Rojas, I., Pomares, H., Bernier, J.L., Ortega, J., Pino, B., Pelayo, F.J., Prieto, A.: Time series analysis using normalized PG-RBF network with regression weights. Neurocomputing 42, 267–285 (2002)
[8] Billings, S.A., Wei, H.L.: A New Class of Wavelet Networks for Nonlinear System Identification. IEEE Transactions on Neural Networks 16(4), 862–874 (2005)
[9] Hong, X., Chen, S.: M-estimator and D-optimality model construction using orthogonal forward regression. IEEE Transactions on Systems, Man and Cybernetics, Part: B 35(1), 155–162 (2005)
[10] Chen, S., Hong, X., Harris, C.J., Sharkey, P.M.: Sparse Modeling Using Orthogonal Forward Regression with PRESS Statistic and Regularization. IEEE Transactions on Systems, Man and Cybernetics, Part: B 34(2), 898–911 (2004)

Spectral Clustering on Manifolds with Statistical and Geometrical Similarity

Yong Cheng[1] and Qiang Tong[2]

[1] Department of Computer Sciences,
Beijing University of Chemical Technology
chengyong@ieee.org
[2] School of Information Technology,
University of International Business and Economics
tongqiang@uibe.edu.cn

Abstract. The problem of clustering data has been driven by a demand from various disciplines engaged in exploratory data analysis, such as medicine taxonomy, customer relationship management and so on. However, Most of the algorithms designed to handle data in the form of point clouds fail to cluster data that expose a manifold structure. The high dimensional data sets often exhibit geometrical structures which are often important in clustering data on manifold. Motivated by the fact, we believe that a good similarity measure on a manifold should reflect not only the statistical properties but also the geometrical properties of given data. We model the similarity between data points in statistical and geometrical perspectives, then a modified version of spectral algorithm on manifold is proposed to reveal the structure. The encouraging results on several artificial and real-world data set are obtained which validate our proposed clustering algorithm.

Keywords: Clustering; Manifold Learning; Spectral Clustering.

1 Introduction

Clustering data with similar features into clusters has been studied in a wide range of literatures and many effective algorithms have been proposed[1]. Despite the success they fail to cluster data that expose a manifold structure, like speech, motion and images, that generally exist in the form of paths in a high-dimensional space. Manifold learning is a learning scheme that characterizes a possibly non-linear manifold on which the data would lie. Popular manifold learning techniques include Locally Linear Embedding (LLE)[2], Hessian LLE[3], ISOMap[4], Laplacian Eigenmaps[5] and so on.

Differing from the manifold learning that primarily discovers a manifold embedding of input data, manifold clustering attempts to partition a set of data into several different clusters each of which contains data points originating from a separate low-dimensional manifold. There exist several works that cluster the data on manifolds. In [6], R. Souvenir et al. presented an approach to factor low-rank manifold of data that originate from multiple, intersecting low-dimensional

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 422–429, 2010.

manifolds. In their method, two novel technical contributions are highlighted: node-weighted multidimensional scaling and a fast algorithm for weighted low-rank approximation for rank-one weight matrices. Q. Guo et al. proposed a manifold clustering approach with energy minimization strategy [7]. In which an energy function has been defined by weighted components on Euclidean distance between two consecutive and discrete curvature of manifolds. The function then be minimized by tabu search to find the locally optimal sequence of the data. Finally, clusters are generated by breaking the sequence and merging some isolated points. It is worthy of noting that the approach only works on 2-D and 3-D data space. In [8], R. Haralick et al. described a new cluster model LMCLUS which is based on the concept of linear manifolds. In order to detect clusters embedded in lower dimensional linear manifolds, LMCLUS uses the strategy of hierarchical-divisive procedure and random projection via sampling and histogram thresholding to construct trial linear manifolds of various dimensions.

In this paper, we are motivated by the fact that high dimensional data sets often exhibit geometrical structures which are important to be considered in clustering data on manifold. We believe that a good similarity measure on a manifold should reflect not only the statistical properties but also the geometrical properties of data sets. We then propose a manifold clustering algorithm SCM which uses spectral analysis to drive the clusters and evaluate the proposed algorithm on several synthetic and real-world data set, the results obtained indicate the improvement in the model quality and give additional insights into the data.

The remainder of this article is organized as follows. The problem of manifold clustering is formulated in Section 2. In Section 3, we first proposed a variant of spectral clustering in which the graph is constructed with the statistical and geometrical similarity, then, the algorithm SCM is described in details. The experimental results on synthetic and real-world are reported in Section 4. Finally, Section 5 concludes the paper.

2 Problem Formulation

Generally speaking, the goal of manifold clustering is to find clusters with an intrinsic dimensionality that is much smaller than the dimensionality of the data set. The problem of clustering data on multiple complex manifold structure can be described as follows: Suppose a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ that derived from S intersecting manifolds, where $\mathbf{x}_i \in \mathbb{R}^d$, n is the size of X, d is the dimension and S is the possible number of manifold. The goal of clustering is to partition the data into S clusters each of which corresponds to a manifold. The output of clustering generally requires a set of labels $C = \{c_1, c_2, \cdots, c_n\}$, where $c_i \in \{1, 2, \cdots, S\}, 1 \leq i \leq n$.

3 SCM: Spectral Clustering on Manifold

3.1 Motivation

The basic goal in clustering analysis is to group data objects with similar features together. In manifold clustering, most existing algorithms utilize certain

traditional similarity measure that emphasizes the statistical properties in given data set. However, we believe that, in additional to statistical properties, high dimensional data sets often exhibit geometrical structures which are often important to be considered in clustering data on manifold: similar data samples should have similar geometrical properties. Furthermore, a good similarity measure on a manifold should reflect not only the statistical properties but also the geometrical properties of given data set, which is the direct motivation of proposed approach in this paper. In our approach, the statistical property of similarity is often implemented with nearest neighborhood technique and the geometrical property of given data sample is measured with data objects whose distances are smaller than a threshold value ε.

3.2 Modeling the Similarity

As introduced in the above, we consider the statistical properties and geometrical properties to model the similarity between data objects. Like most of the traditional clustering methods, the k nearest neighborhood techniques is chosen to calculate the similarity in the perspective of statistical properties. In addition, the similarity in geometrical property of given data object is modeled with ε neighborhood in our consideration. The k nearest neighborhood and ε neighborhood are defined as follows:

Definition 1 (k-neighborhood). *$\mathcal{N}_i(k)$ is a set of data points in dataset X that contains k nearest points of $\mathbf{x}_i$.*

Definition 2 (ε-neighborhood). *$\mathcal{N}_i(\varepsilon)$ is a set of data points in dataset X that contains points satisfying $\|\mathbf{x}_i - \mathbf{x}_j\| < \varepsilon$, i.e.*

$$\mathcal{N}_i(\varepsilon) = \{\mathbf{x}_j | \|\mathbf{x}_i - \mathbf{x}_j\| < \varepsilon, \mathbf{x}_j \in X\} \tag{1}$$

k-neighborhood and ε-neighborhood describe the local relationship of a data point $\mathbf{x}_i$ in statistical and geometrical aspects respectively. Thus, we can define the k-neighborhood and ε-neighborhood graph as follows:

Definition 3 (k-neighbor Graph). *Given a set of points $X = \{\mathbf{x}_i, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, and the k-neighborhood of each point $\mathbf{x}_i, 1 \leq i \leq n$. A weighted undirected graph $G_k = \langle V_k, E_k \rangle$ is constructed from the given data set X where,*

- *$V_k = X$, i.e. that each vertex in the G_k corresponds a data object in X;*
- *$E_k = \{\mathbf{x}_i\mathbf{x}_j | \mathbf{x}_i \in \mathcal{N}_j(k)\}$, that is two data instances $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if and only if one is in the k-neighborhood of the other;*
- *The weight w_{ij}^k of edge connecting $\mathbf{x}_i$ and $\mathbf{x}_j$ is the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$;*

Definition 4 (ε-neighbor Graph). *Given a set of points $X = \{\mathbf{x}_i, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, and the ε-neighborhood of each point $\mathbf{x}_i, 1 \leq i \leq n$. A weighted undirected graph $G_\varepsilon = \langle V_\varepsilon, E_\varepsilon \rangle$ is constructed from the given data set X where,*

- $V_\varepsilon = X$, *i.e. that each vertex in the* G_ε *corresponds a data object in* X*;*
- $E_\varepsilon = \{\mathbf{x}_i\mathbf{x}_j | \mathbf{x}_i \in \mathcal{N}_j(\varepsilon)\}$*, that is two data instances* $\mathbf{x}_i$ *and* $\mathbf{x}_j$ *are connected if and only if one is in the* ε*-neighborhood of the other;*
- *The weight* w_{ij}^{ε} *of edge connecting* $\mathbf{x}_i$ *and* $\mathbf{x}_j$ *is the weighted similarity between* $\mathbf{x}_i$ *and* $\mathbf{x}_j$*;*

In general, there exists three possible choices to weight the edge in graph G. The simplest method for building the weighted graph is the *binary* weighting approach, the second approach is *Gaussian Kernel(GK)* which is often applied in many application, the final method to estimate the edge is given by the locally linear embedding technique and the weight w_{ij} can be calculated by solving an optimization problem [2].

Definition 5 (Similarity Graph). *Given a set of points* $X = \{\mathbf{x}_i, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$*, and the* k*-neighborhood and* ε*-neighborhood of each point* $\mathbf{x}_i, 1 \leq i \leq n$*. The* k*-neighbor graph* $G_k = \langle V_k, E_k \rangle$ *and* ε*-neighbor graph* $G_\varepsilon = \langle V_\varepsilon, E_\varepsilon \rangle$ *can be constructed according to definition (3) and definition (4). Thus, a weighted graph that combines the statistical and geometrical similarity will be created as* $G = \langle V, E \rangle$*, where*

- $V = X$*, i.e. that each vertex in the* G *corresponds a data object in* X*;*
- $E = \{E_k \cup E_\varepsilon\}$*, that is two data instances* $\mathbf{x}_i$ *and* $\mathbf{x}_j$ *are connected if one is in the* k*-neighborhood or* ε*-neighborhood of the other;*
- *The weight* w_{ij} *of edge connecting* $\mathbf{x}_i$ *and* $\mathbf{x}_j$ *is the linearly combined weighted similarity between* $\mathbf{x}_i$ *and* $\mathbf{x}_j$*, and*

$$w_{ij} = \alpha w_{ij}^{k} + (1 - \alpha) w_{ij}^{\varepsilon} \quad \alpha (0 \leq \alpha \leq 1) \text{ is the coefficient} \tag{2}$$

As mentioned above, the graph $G = \langle V, E \rangle$ considers the two local relationship of a data point $\mathbf{x}_i$, i.e. the k-neighborhood and ε-neighborhood which are incorporated into a unified graph representation via linear combination. It is worthy of noting that the graph is not symmetrical, like most approaches in the literature, here, we also use the symmetrical graph by $W' = \frac{1}{2}(W + W^\top)$. Without loss the generality, W denotes the symmetrical similarity in the remainder of the paper. After the data points and local relationship representation in graph is obtained, we can derive the clusters using the spectral analysis.

3.3 SCM Algorithm

Spectral clustering works by detecting the clusters in data set via analyzing the eigenvectors of graph Laplacian. In other words, the multiplicity k of the eigenvalue 0 of unnormalized graph Laplacian L or normalized graph Laplacian L_{norm} equals the number of connected components in the graph. The graph Laplacian L and L_{norm} are computed as follows,

$$L = D - W \tag{3}$$

$$L_{norm} = D^{-1/2} L D^{-1/2} \tag{4}$$

where D is the degree matrix of W, $D_{ii} = \sum_j w_{ij}, 1 \leq i \leq n$.

Following the general framework of spectral clustering, we can derive our spectral clustering variant that clusters data on a manifold considering the statistical and geometrical similarity. The proposed algorithm is described in Figure (1).

Algorithm 1. SCM: Spectral Clustering Algorithm on Manifold

Input: Data set X, statistical and geometrical neighborhood k, ε, the bandwidth of kernel σ, the number of clusters K;

Output: Clusters $C_1, C_2, \cdots, C_K$;

1 Construct the k-neighborhood graph with statistical similarity, Let W_k be its weighted matrix;

2 Construct the ε-neighborhood graph with geometrical similarity, Let W_ε be its weighted matrix;

3 Construct a similarity graph by considering the statistical and geometrical similarity with Equation (2). Let $W = \alpha W_k + (1-\alpha)W_\varepsilon$ be its weighted matrix;

4 Compute the unnormalized Laplacian L or the normalized Laplacian L_{norm};

5 Compute the first K generalized eigenvectors $u_1, u_2, \cdots, u_K$ of the generalized eigen problem $Lu = \lambda Du$;

6 Let $U \in \mathbb{R}^{n \times K}$ be the matrix containing the vectors $u_1, u_2, \cdots, u_K$ as columns;

7 For $i = 1, \cdots, n$, let $y_i \in \mathbb{R}^K$ be the vector corresponding to the i-th row of U;

8 Cluster the points $(y_i)_{i=1,\cdots,n}$, in $\mathbb{R}^K$ with the k-means algorithm into clusters $C_1, \cdots, C_K$;

In the first step, the complexity for compute statistical k-neighborhood for all points is $O(n^3 \cdot \log(n))$, while the complexity is $O(n^3)$ for the geometrical ε-neighborhood, where n is the number of data points. The later three steps are implemented in a spectral clustering algorithm, the time complexity of the implemented algorithm depends on the complexity of the eigenvalue decomposition algorithm is about $O(n^3)$, where n is the number of rows/columns. In most cases, it is possible to reduce the time complexity, since the algorithm needs certain eigenvectors only (which are corresponding to smallest or largest eigenvalues in magnitude). Thus, the proposed algorithm has about the same time complexity with the classic spectral clustering.

4 Experimental Results

We experiment the proposed algorithm on several synthetic and real-world data from UCI machine-learning repository [9]. Our spectral is in fact a general variant of classic spectral clustering. There have several parameters need to be set. In all experiments, we select the Gaussian Kernel to measure similarity between two data points and the bandwidth of kernel σ is set to 2. In addition, we treat the statistical similarity and geometrical similarity equally, so the coefficient α is set to 0.5. For the purpose of comparing performance of clustering, RandIndex is selected to evaluate the performance of our algorithm. In each experiment, we run the algorithm on each data set for 10 times, then the mean RandIndex value is calculated.

4.1 Synthetic Data

We select two synthetic data sets, namely "twomoons" and "spiral" data for evaluating our proposed algorithm. The "twomoons" and and "spiral" data set have become the standard benchmarks in numerous other manifold related experiments. Note that these synthetic data cannot be clustered in a meaningful way by certain methods that assume the data form a compact shape. The original data set of "twomoons" and "spiral" are illustrated in Figure (1) and (5), respectively.

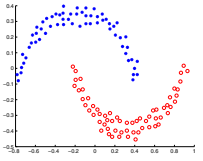


Fig. 1. Two moons

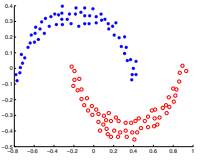


Fig. 2. $k = 4, \varepsilon = 0.2$

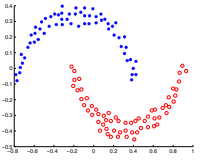


Fig. 3. $k = 1, \varepsilon = 0.2$

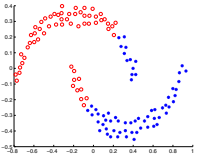


Fig. 4. $k$means

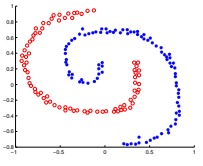


Fig. 5. Spiral

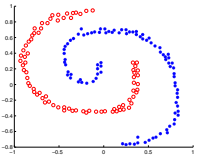


Fig. 6. $k = 4, \varepsilon = 0.2$

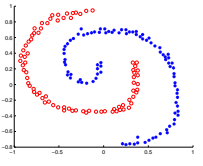


Fig. 7. $k = 1, \varepsilon = 0.2$

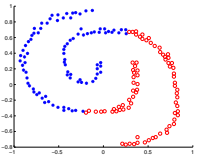


Fig. 8. $k$means

The clustering results of our proposed algorithm are shown in figure (2) to (3) for twomoons data and figure (6) to (7) for spiral data. We can see our algorithm work very well with different configuration of parameters. Figure (4) and (8) are the results that the k-means algorithm returns. It is obvious that our algorithm can find much more meaningful clusters than the classic k-means for the data on the manifold.

4.2 Real-World Dataset

We also test our algorithms on several real-world data sets from the UCI machine-learning repository [9]. Four data, namely "Soybean", "Vowel", "Iris" and "Zoo" are selected to measure the performance with RandIndex. All data are multi-attirubte and multi-class that are considered to situate on the manifolds. The basic information of these data sets is described in the Table (1).

In our experiment, different configuration of parameters are tested to compare the performance improved. Four k-neighborhood, i.e. $k = 0, 10, 20, 30$ nearest neighbors are constructed as the statistical similarity, and four ε-neighbors, i.e.

Table 1. Data set description from UCI

Name	Num of Insts	Num of Attrs	Num of Disc.	Num of Cont.	Num of Class
Soybean	683	36	36	0	19
Vowel	990	14	4	10	11
Iris	150	5	4	1	3
Zoo	101	18	17	1	7

$\varepsilon = 0, 0.5, 2, 5$ geometrical neighborhoods are constructed as the geometrical similarity. Thus, our proposed algorithm is tested on the possible statistical and geometrical similarity of 15 combination to measure the clustering performance, the coefficient is set to 0.5 in all experiments (except for $k = 0$ and $\varepsilon = 0$). The experiment results are shown in the Table. (2) to (5).

Table 2. Soybean data

k	$\varepsilon = 0$	$\varepsilon = 0.5$	$\varepsilon = 2$	$\varepsilon = 5$
$k = 0$	N/A	.8307	N/A	.7552
$k = 10$	.8334	.8338	.8494	.8099
$k = 20$	.7760	.7761	.7779	.7775
$k = 30$	.7994	.8030	.6860	.8503

Table 3. Vowel data

k	$\varepsilon = 0$	$\varepsilon = 0.5$	$\varepsilon = 2$	$\varepsilon = 5$
$k = 0$	N/A	N/A	N/A	.8286
$k = 10$	.8011	.8160	.8141	.8277
$k = 20$	.8090	.7951	.8157	.8312
$k = 30$	.8164	.8342	.8359	.8280

Table 4. Iris data

k	$\varepsilon = 0$	$\varepsilon = 0.5$	$\varepsilon = 2$	$\varepsilon = 5$
$k = 0$	N/A	.7766	.8797	.8797
$k = 10$	.8759	.8859	.8797	.8797
$k = 20$	.9017	.9055	.8797	.8797
$k = 30$	.9124	.9124	.8787	.8737

Table 5. Zoo data

k	$\varepsilon = 0$	$\varepsilon = 0.5$	$\varepsilon = 2$	$\varepsilon = 5$
$k = 0$	N/A	.6945	.6669	.9002
$k = 10$	.8996	.8976	.8996	.8996
$k = 20$	.7826	.7869	.8994	.8994
$k = 30$	.8924	.8994	.8994	.8994

From these figures, we can see the proposed algorithm can certainly improve the performance of clustering on manifolds. For the "soybean" data, the best clustering results is obtained when $k = 30, \varepsilon = 5$. Our algorithm uses the statistical and geometrical similarity to describe the local neighborhood relationship and outperform classic spectral clustering (using k-neighborhood or ε-neighborhood) and k-means algorithm whose randIndex is .5427.

5 Conclusions and Future Work

In this paper, we proposed a variant of spectral clustering that perform well on several data sets. Our work is motivated by the fact that similar data points in high dimensional data sets often exhibit similar geometrical structures. Differing from most existing algorithms that emphasize the statistical properties in given

data set, we consider a good similarity measure on a manifold should reflects not only the statistical properties but also the geometrical properties of given data set. We use the k-neighborhood and ε-neighborhood to model the statistical and geometrical aspects respectively and derived the new variant of spectral clustering. An extensive experiments have been conducted to test the quality of the clusters produced by proposed algorithm on a varied collection of artificial and real-world data set. The results indicate promising performance that validate our proposed algorithm.

Acknowledgments

This work is supported by the young teacher project in BUCT under contract (No. QN0824). Qiang Tong thanks the 211 project in UIBE under contract (No. 32046).

References

1. Han, J., Kamber, M.: Data Mining - Concepts and Techniques, 2nd edn. China Machine Press, Beijing (2007)
2. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
3. Donoho, D.L., Grimes, C.: Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. In: Falkow, S. (ed.) Proceedings of the National Academy of Sciences of the United States of America, May 2003, vol. 100, pp. 5591–5596 (2003)
4. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)
5. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation 15(6), 1373–1396 (2003)
6. Souvenir, R., Pless, R.: Manifold clustering. In: Proceedings of the 10th International Conference on Computer Vision (ICCV 2005), October 17-21, vol. 1, pp. 648–653 (2005)
7. Guo, Q., Li, H., Chen, W., Shen, I.F., Parkkinen, J.: Manifold clustering via energy minimization. In: Proceedings of the 6th International Conference on Machine Learning and Applications (ICMLA 2007), December 13-15, pp. 375–380. IEEE Computer Society, Los Alamitos (2007)
8. Haralick, R., Harpaz, R.: Linear manifold clustering. In: Perner, P., Imiya, A. (eds.) MLDM 2005. LNCS (LNAI), vol. 3587, pp. 132–141. Springer, Heidelberg (2005)
9. Asuncion, A., Newman, D.: UCI machine learning repository (2009)

A Supervised Fuzzy Adaptive Resonance Theory with Distributed Weight Update

Aisha Yousuf and Yi Lu Murphey

University of Michigan – Dearborn
Department of Electrical and Computer Engineering,
4901 Evergreen Road,
Dearborn, MI 48128 USA
ayousuf@umich.edu

Abstract. The Fuzzy Adaptive Resonance Theory is an unsupervised clustering algorithm that solves stability plasticity dilemma. The existing winner-take-all approach to updating weights in Fuzzy ART has two flaws: (i) it only updates one cluster while an input might belong to more than one cluster and (ii) the winner-take-all approach is costly in training time since it compares one weight to the input at a time. We propose an algorithm that compares all weights to the input simultaneously and allows updating multiple matching clusters that pass the vigilance test. To mitigate the effects of possibly updating clusters belonging to the wrong class we introduced weight scaling depending on the "closeness" of the weight to the input. In addition, we introduced supervision to penalize the weight update for weights that have the wrong class. The results show that our algorithm outperformed original Fuzzy ART in both classification accuracy and time consumption.

Keywords: Fuzzy ART, supervised learning, clustering.

1 Introduction

Fuzzy Adaptive Resonant Theory (Fuzzy ART) is an unsupervised neural network that answers the stability-plasticity dilemma. The network is able to learn new patterns (plastic) while retaining the information from previously learned knowledge (stable) [1], [2], [3]. One of the major limitations of the Fuzzy ART is the winner take all approach, which allows only one prototype (weight) to be updated at a time [4]. However, a training point might have the same class as more than one prototype. In order to maximize the learning potential of the algorithm, all the prototypes that meet the threshold requirements should be updated.

In the conventional Fuzzy ART if the "closest match" prototype to the input that passes the vigilance test belongs to the wrong class, then by updating only that one weight minimizes the separation between the classes. The advantage of the proposed method is that it addresses this shortcoming by distributed weight updating. Out of multiple weights that get updated, some might belong to the correct class. The main disadvantage of having a distributed weight update is that the prototypes that are not

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 430–435, 2010.

the closest match and pass the vigilance test are updated even though the input is from a different class. The effect of distributing to the wrong classes is mitigated in the proposed approach through weight scaling and supervision. Updating weight for wrong classes differently maximizes the separation between classes.

This paper is organized as follows. In section 2, we discuss the algorithms including the original Fuzzy ART and proposed Supervised Distributed Fuzzy ART (SDF ART) algorithm. The experiments and results are discussed in section 3. We conclude and describe our future plans in section 4.

2 Algorithms

2.1 Original Fuzzy ART

Fuzzy ART [1], [2], [3] is based on combination of the ART1 network and fuzzy logic and uses the fuzzy operators min(∧), and max(∨). The input to the network, I, is normalized by appending the actual input, A, to its complement 1-A.

The normalized input pattern I is used for computing y as shown in Eq. 1. w matrix contains all the weight (prototype) vectors. Alpha, which is chosen by the user, is the conservation limit, which when small, minimizes recoding during learning.

$$y_j = \left|I \wedge w_j\right| \Big/ \left(\alpha + \left|w_j\right|\right). \tag{1}$$

The node that yields the largest value of y is selected as the winner. The winner, which is denoted by y_J with J as the winning node index, has to pass the vigilance test shown in Eq. 2. ρ, a value between 0 and 1, is a vigilance parameter set by the user. k is the number of existing prototypes.

$$\rho \le \left|I \wedge w_J\right| \Big/ k \ . \tag{2}$$

If the test is passed, resonance occurs. The input I joins cluster J and the winning prototype vector, w_J, is updated using Eq. 3. β is the learning rate and ranges between 0 and 1.

$$w_J^{new} = \beta\left(I \wedge w_J^{old}\right) + \left(1-\beta\right) w_J^{old} \ . \tag{3}$$

If the test is failed, the winning prototype is inhibited. The next highest value of y from Eq. 1 is selected as the winner and passed on to the test in Eq. 2. This process continues until the test is passed or if all the components of y are inhibited. In the latter case, a new cluster is formed and the input is the new cluster.

2.1 Kondadadi Distributed Weight Update

A recent paper [6] discusses the areas of improvement for Fuzzy ART. One of areas discussed is that distributed weight update is a much needed improvement. However, very little work has been done on this topic. Kondadadi et al [4] show the distributed weight update by updating every single prototype that matches the input. However, they fail to take into account that while these prototypes pass the vigilance test, they

might not all belong to the correct class. We, therefore, impose a criterion that scales the weight update by the "closeness" of the input to the prototypes. Furthermore, we integrated supervised learning to penalize the weight update for the incorrect class.

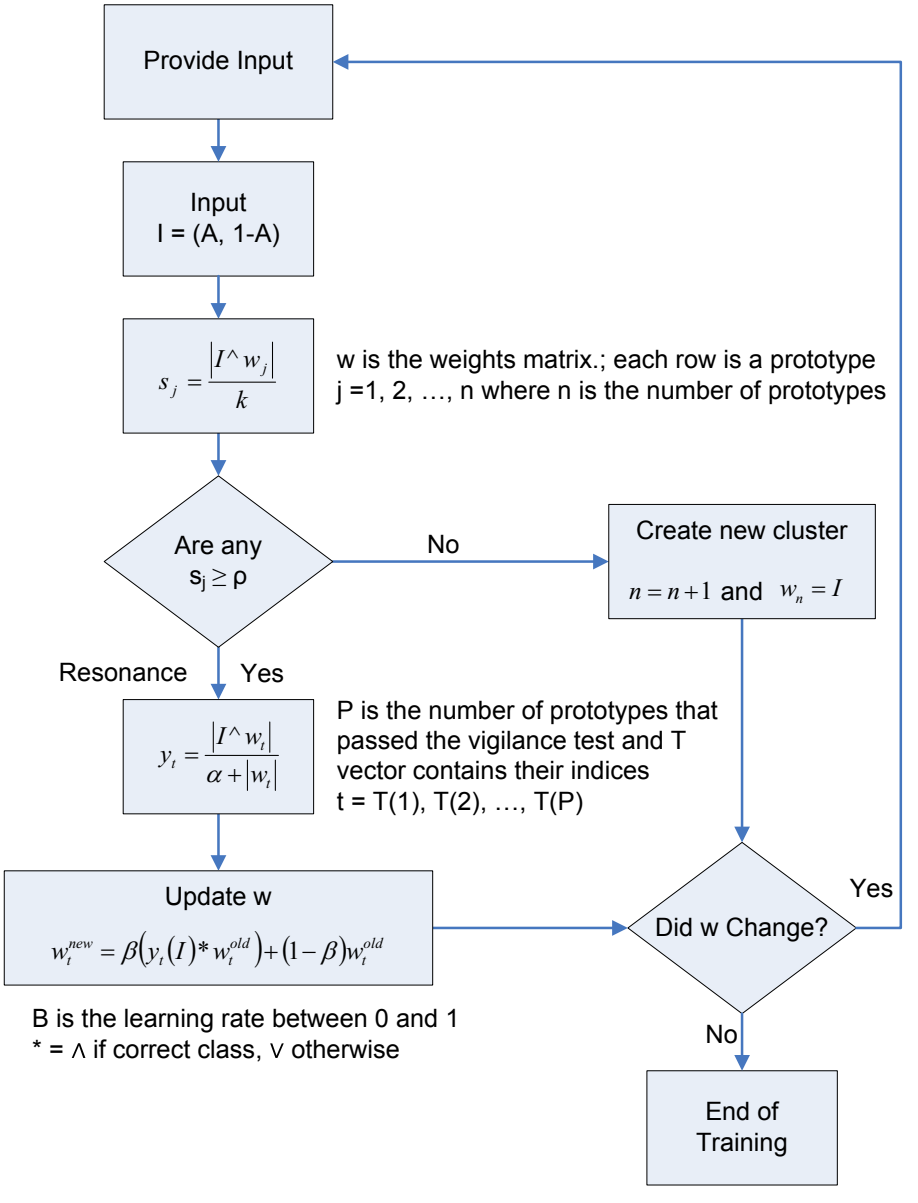


Fig. 1. SDF ART Structure

2.3 Supervised Distributed Fuzzy ART

The proposed SDF ART is shown in Fig. 1. Unlike original Fuzzy ART it does not select the "closest match" winner. Instead it goes directly to the vigilance test which selects all weights that meet the threshold requirements. If no prototypes satisfy the vigilance criteria, then a new cluster is formed with the input as the prototype. This makes the algorithm more time efficient as it does not have to go through the loop to find one winning cluster at a time.

While distributing input to multiple prototypes, since it is possible to update other weights that don't match the class, the weight updates are scaled. The weights are updated using the same "closeness" criteria used in Original Fuzzy ART to find the winner shown in Eq. 1. In addition, the supervised approach was added to penalize the weight update in case it was the wrong class. This was accomplished by performing the weight update with an operator, *, which is min($\wedge$) if it is the correct class, and max($\vee$) if it belongs to the wrong class. The formula for the updated weight is shown in Eq. 4, where * is the supervision operator and y_t is the weight scale factor.

$$w_t^{new} = \beta\left(y_t(I) * w_t^{old}\right) + (1-\beta) w_t^{old} \tag{4}$$

During the classification stage, there are some prototypes that were formed using inputs from multiple classes. These prototypes are assumed to belong to the class that had the majority number classified inputs to that prototype.

3 Experiments and Results

To evaluate the proposed algorithm we applied the SDF ART, and original Fuzzy ART for comparison, to Ionosphere dataset from the UCI Machine Learning Repository [5]. The data contains RADAR returns from the ionosphere for classification into good or bad returns. The entire dataset consists of 351 feature vectors, out of which we used 234 points for training and 117 for testing.

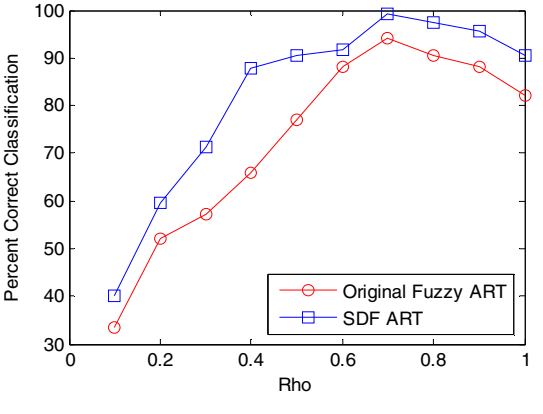


Fig. 2. Percent Correct Classifications

Experiments were conducted for studying the changes in the parameters. Since Fuzzy ART is affected by the order of inputs, the order of inputs was kept consistent during the parameter tests to get a fair comparison of effects with changes in control parameters. The same input order was kept for original Fuzzy ART and SDF ART. The percent correct classification rates for Fuzzy ART and SDF ART algorithms are shown in Fig 2. It can be observed from Fig. 2 that SDF ART always outperforms the original Fuzzy ART.

Table 1 shows the number of cycles it takes to converge with the change in beta (the learning rate). With larger beta both Fuzzy ART and SDF ART take fewer cycles, whereas they take longer with smaller beta. However, it can be observed that regardless of what beta is, SDF ART always takes less cycles to converge. In addition, the SDF ART converged in less than one third of the time that it took original Fuzzy ART to converge.

Table 1. Number of Cycles

Beta (β)		0.001	0.25	0.5	0.75	1
# of cycles	Fuzzy ART Original	879	767	542	414	134
	SDF ART	871	760	535	407	124

Table 2. Number of clusters with change in rho

Rho (ρ)		.001	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
# of clusters	Fuzzy ART	1	8	16	26	49	62	71	85	98	107	115
	SDF ART	1	6	14	21	46	57	64	82	95	102	106

Table 3. Percent Correct Classification with Change in Input Order

	Test 1	Test 2	Test 3	Test 4	Test 5
% Correct Classification	99.15%	97.44%	96.56%	99.15%	97.44%

The number of prototypes formed with changing values of rho is shown in Table 2 for both the original Fuzzy ART and SDF ART algorithms. On average, the SDF ART has approximately 4 prototypes less than original Fuzzy ART. Having fewer prototypes gives SDF ART an added advantage since it allows faster classification during testing.

Another test done on the modified Fuzzy ART was to study the variations in the accuracy with change in input order. Table 3 depicts the classification results with different input orders. It was observed that even with the change in input order the results were fairly consistent and within 3% of each other. This shows that the SDF ART algorithm is able to produce consistent results regardless of input orders.

Lastly, we compared our distributed weight update approach to the one proposed by Kondadadi et al [4]. Fig. 3 shows our percent correct classification results vs. theirs. It can be observed that our approach performs slightly better. However, their algorithm converges faster taking about 3/4 of the time it takes our algorithm to converge.

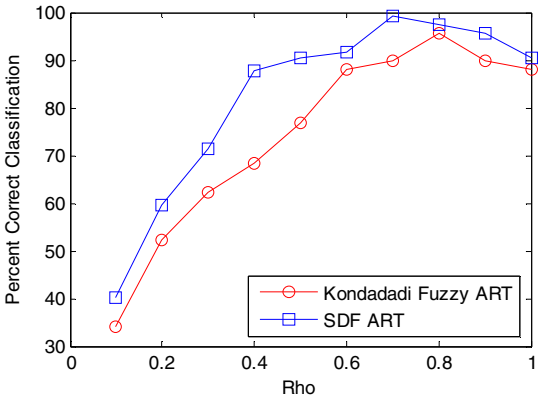


Fig. 3. Percent Correct Classification for SDF ART and Kondadadi approach

4 Conclusions and Future Work

It was observed that the SDF ART outperformed the Original Fuzzy ART in two aspects. It had better classification performance, and it takes less time during training and testing. It is expected that when the dataset is larger, the performance of SDF ART would be even better than original Fuzzy ART. In addition, it is expected that the time difference for training and testing between the SDF ART and fuzzy ART would be even greater. It was also observed by comparing results with Kondadadi approach that weight scaling performs better when it comes to classification.

For future work, we plan on using the INRIA pedestrian dataset with 9000+ points to classify pedestrians versus the background using the proposed approach. In addition, we will test this on datasets that have more than 2 classes. Lastly, we will explore additional methods of determining which class the cluster belongs to instead of selecting majority class since the weight updates are also scaled by the closeness.

References

1. Burwick, T., Joublin, F.: Optimal Algorithmic Complexity of Fuzzy ART. Neural Processing Letters 7, 37–41 (1998)
2. Carpenter, G., Grossberg, S., Rosen, D.: Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System. Neural Networks 4, 759–772 (1991)
3. Huang, J., Georgiopoulos, M., Heileman, G.: Fuzzy ART properties. Neural Networks 8, 203–213 (1995)
4. Kondadadi, R., Kozma, R.: A Modified Fuzzy ART for Soft Document Clustering. In: Proceedings of the 2002 International Joint Conference on Neural Networks, pp. 2545–2549 (2002)
5. UCI Machine Learning Repository, `http://archive.ics.uci.edu/ml/`
6. Wunsch II, D.: ART Properties of Interest in Engineering Applications. In: Proceedings of the 2009 International Joint Conference on Neural Networks, pp. 3380–3383 (2009)

A Hybrid Neural Network Model Based Reinforcement Learning Agent

Pengyi Gao[1,*], Chuanbo Chen[1], Kui Zhang[2], Yingsong Hu[1], and Dan Li[1]

[1] School of Computer Science and Technology, HuaZhong University of Science and Technology,WuHan 430074, China
[2] School of Mechanical, Aerospace and Civil Engineering, University of Manchester, Oxford Road, Manchester, M13 9PL, United Kingdom
Pengyi_gao@mail.hust.edu.cn

Abstract. In this work, a hybrid neural network model (HNNM) is proposed, which combines the advantages of genetic algorithm, multi-agents and reinforcement learning. In order to generate networks with few connections and high classification performance, HNNM could dynamically prune or add hidden neurons at different stages of the training process. Experimental results have shown to be better than those obtained by the most commonly used optimization techniques.

Keywords: Multi-agent, Reinforcement learning, Neural networks, Optimization, Genetic algorithm.

1 Introduction

It is one of the most important issues over many years for researchers how to design an optimal and adaptive architecture for artificial neural networks (ANNs), especially hidden neurons and connections weights. There have been many approaches to adjust hidden neurons of ANNs, such as constructive or pruning, constructive and pruning, evolutionary and hybrid algorithms [1,2,3,4,5], etc. But, all of these methods have a common weakness, i.e. not able to adjust automatically networks architecture.

In order to dynamically adjust hidden neurons, Islam presented an adaptive merging and growing algorithm (AMGA), which could autonomously merge and add hidden neurons of ANNs [6]. However, there are some problems in the algorithm, for example, significance of each hidden neuron depends on experience, and specified parameters, such as significance threshold, can not be modified automatically. It is also difficulty to compute the correlation coefficient between selected neuron and other neuron, Additionally, BP training algorithm is easy to be trapped into local minima [7]. Considering the factors in AMGA, a hybrid learning model based on reinforcement learning (RL) agent is proposed in this work, where the neural networks consisting of three layers are trained by NN agent using GA algorithm, while training parameters will be chosen by RL agent, which include significance of hidden neurons, epochs, errors, initial weights, and hidden neurons, etc.

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 436–443, 2010.

The paper is organized as follows. Section 2 introduces proposed hybrid neural network learning model, and describes the function of main agents. In section 3, learning algorithm for NN agent is described in detail. Section 4 offers an insight into learning algorithm of RL agent. Experimental results are discussed in section 5 and the conclusions and a look to future work is put forward in the last.

2 Architecture of Hybrid Learning Model

In the model, we design a multi-agent platform, which consists of two containers, main container and agent container, shown in Fig.1. Two agents, NN agent and RL agent live in containers which provide a run-time and the services for agents. NN agent is responsible of pruning or adding hidden neurons and modifying weights of ANN by GA. RL agent receives the reward of NN agent and adjusts training parameters to NN agent according to the reward values and expert knowledge. Agents cooperate with each other to finish a given tasks [8,9,10].

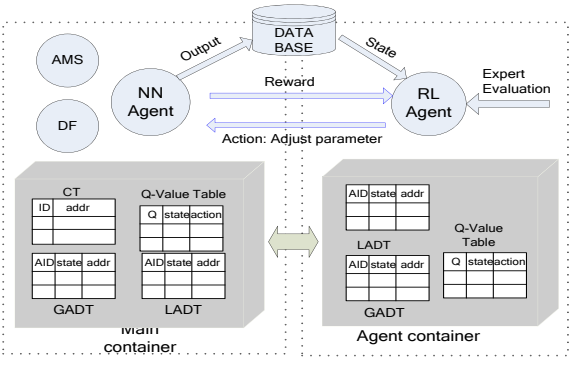


Fig. 1. The proposed learning model based multi-agent

The main container has the following special responsibilities:

- Managing the container table (CT).
- Managing agent descriptor table GADT and LADT.
- Managing Q-value table (QVT) for RL agents.

The AMS and the DF provide the agent management and white page service, and the default yellow page service of the platform, respectively.

3 Learning Algorithm for NN Agent

The aim of NN agent is to find a global best solution by GA algorithm. A solution vector is corresponding to an ANN, which consists of the number of neurons for each layer, ID of each hidden neuron, and their weights.

3.1 Algorithm Procedure

The developed algorithm for NN agent consists of the following steps:

Step 1: Generate randomly an initial population, which consists of variable length real-valued chromosomes, and each chromosome is corresponding to an ANN. Initialize the number of hidden neurons H and all connection weights at random. Set training epoch μ_i=0, i = 1, 2,. . .,H, for each hidden neuron h_i.
Step 2: The evolutionary process begins by applying the genetic operators (fitness, selection, crossover and mutation).
Step 3: Compute the objective function, that is sum of squared error (SSE) for each trained ANN and assign a probability for each solution according to SSE.
Step 4: Store the ANN architecture with the lowest objective value.
Step 5: Select a pair parent to perform crossover by the way of binary tournament [11], in order to reproduce the offspring that will be reinserted into the population replacing the solution with the smallest fitness.
Step 6: If the best offspring ANN has a lowest SSE, replace the last stored ANN.
Step 7: Generate randomly a new offspring by mutation operator in order to avoid being trapped into local minima.
Step 8: Execute steps 2–7 again until GA converges or maximum number of generations is reached.
Step 9: Compare SSE for the current best solution with the previous. If larger than the previous, then restore the previous network.
Step10: Increase epoch: $\mu_i = \mu_i + \tau$.
Step11: Save the current solution and the SSE into Database.
Step12: If termination criteria are met, stop the procedure and output the best.
Step13: Select the neurons, which significance η_i received from RL agent is less than threshold η_{th}. If not, go to the Step 16.
Step14: Compute the correlation between the selected hidden neurons and other hidden neurons in the network based on the output of hidden neurons.
Step15: If the pairs are found, then merge them and generate a new population, go to step 2 and retrain the networks.
Step16: If the neuron addition criterion is not satisfied, go to the Step 11.
Step17: Add one neuron to the hidden layer by splitting an existing neuron. The splitting produces two neurons from one neuron. Generate a new population and go to the Step 2 and retrain the networks.

3.2 Objective Function Definition

Objective function is defined by classification error TER, expressed as follows [7]:

$$TER = \frac{100}{\#P} \sum_{y \in P} \varepsilon(y) \tag{1}$$

Where #P is the number of patterns, $\varepsilon(y)$ is the error for the pattern y, which is 0 when true class of the pattern y is same as the desired class, otherwise, 1.

3.3 Correlation Coefficient Definition

Correlation measure can be used to measure linear relationship between variables, and it can also be used to find the correlation between different hidden neurons in the ANN. The correlation coefficient between $\mathbf{h}_i$ and $\mathbf{h}_j$ is defined as [12]:

$$P_d = \frac{cov\left(\mathrm{h}_i, \mathrm{h}_j\right)}{\sqrt{var(\mathrm{h}_i)var(\mathrm{h}_j)}} \tag{2}$$

Where cov is the covariance and var the variance, $\mathbf{h}_i$ and $\mathbf{h}_j$ are the output vector of a selected hidden neuron i and another unselected hidden neuron j, respectively in the training set. Actually, the coefficient is the cosine between output vectors.

3.4 Neuron Merging Criterion

If the correlation between neuron h_a and h_b is greater than the threshold, NN agent merges the two neurons into a new neuron h_m. The weights of h_m are assigned as

$$w_{mi} = (w_{ai} + w_{bi}) / 2 \tag{3}$$

$$w_{jm} = w_{ja} + w_{jb} \tag{4}$$

where w_{ai} and w_{bi} are the connection weights from ith input neuron to h_a and h_b, respectively, while w_{ja} and w_{jb} are the connection weights to jth output, respectively. W_{mi} and w_{jm} are the ith input and jth output connection weights of h_m, respectively [6].

3.5 Neuron Addition Criterion

The weights of the new two neurons created by splitting an existing neuron are calculated as follows:

$$w^1 = (1+\alpha)w \tag{5}$$

$$w^2 = -\alpha w \tag{6}$$

where $\mathbf{w}$ represents the weight vector of the existing neuron, and $\mathbf{w}^1$ and $\mathbf{w}^2$ are the weight vectors of the new neurons. α is a small mutation parameter.

3.6 Termination Criterion

The procedure is terminated when one of the criteria is met as follows:

① The number of successive iterations without any improvement of the objective function value is equal to the maximum.
② The validation error increases for T consecutive times.
③ Specified number of generations is reached.

4 Learning Algorithm for RL Agent

RL agent can complete the task to adjust automatically networks architecture through cooperation with NN agent. The learning process includes two main stages. First, RL agent observes the output of NN agent, and the state of solutions from Database, and explores an action that can yield the maximum reward, based on the former experience associated with the current observation and accumulated reinforcement (reward). Next, the agent takes those actions.

At the beginning, RL agent has nothing about the effect that actions may make. It discovers the actions, such as modifying the significance of hidden neuron according to an explored optimal policy which may bring the highest reward. After a while, the agent gradually takes those actions and then observes the results from NN agent. In fact, the reward can be defined objectively based on the results, or gained subjectively by directly receiving evaluation from the interactive expert.

State definition. Define the states s as a vector consisting of architecture of networks and modified hidden neurons.

Action definition. Define the actions a as a vector consisting of modified significance of hidden neuron, epoch and GA parameters such as population size.

Reward definition. Define the reward r as a vector by:

$$r_i = \frac{1}{TER_i} \tag{7}$$

Where r_i is the reward received at neuron i, TER_i is classification error associated with neuron i, it means that TER_i is evaluated according to its influences on training results when neuron i is merged or added into the ANN.

Reward updating criterion. RL agent uses Q-learning algorithm to learn the optimal action policy. The action value function Q(s,a) in this case satisfies equation as follows:

$$\mathrm{Q(s,a)} \leftarrow \mathrm{Q(s,a)} + \alpha(\mathrm{r} + \gamma(\max_{a'} \mathrm{Q(s',a')} - \mathrm{Q(s,a)}) \tag{8}$$

Where r is a received reward value, α is a learning rate, and γ is the discounting factor. In order to simplify computation, the policy update is performed based on a method called reinforcement comparison [13], which is updated by

$$\overline{r}_i(k+1) = \overline{r}_i(k) + \alpha(r_i(k) - \overline{r}_i(k)) \tag{9}$$

Where α $(0 < \alpha \leq 1)$ is a learning rate, and $\overline{r}_i(k)$ is reference reward of $r_i(k)$ in general. Then update the policy only if the received reward $r_i(k) > \overline{r}_i(k)$

5 Experimental Results

5.1 Testing Effectiveness of Algorithm

In this experiment, in order to evaluate the effectiveness of the proposed method, an empirical study is carried out on Diabetes data sets, which has eight inputs, two outputs, and 768 examples. In the training phase for the classifiers, 768 samples will be randomly divided into two subsets, training set with the 60% of samples and testing set with the remaining 40%.

All input are transformed within the range from 0 to 1. Initial population size is set 20, crossover rate is 0.9, and mutation rate is 0.1.

Figure 3 shows results including TER, and the best number of hidden neurons. TER is less when number of hidden neurons is 5, 10 and 11, respectively.

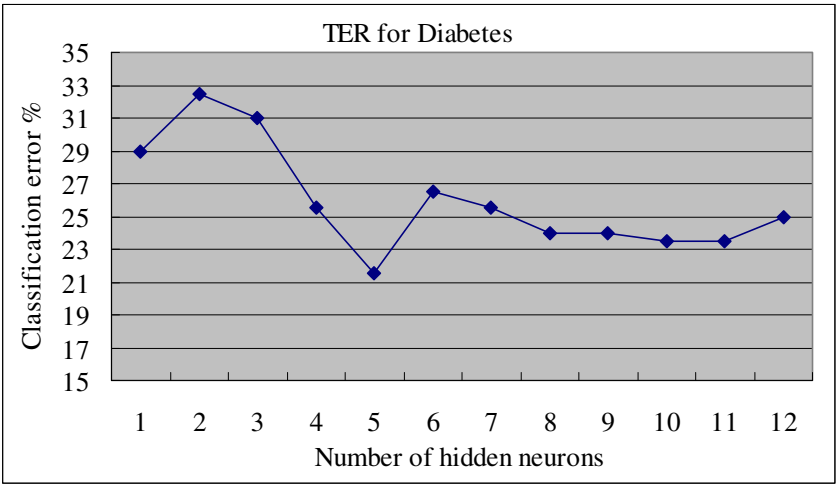


Fig. 2. Compute classification error for different hidden neurons. When the number of hidden neurons is 5, classification error is 21.5%, it is best solution.

5.2 Comparing with Other Algorithms

In order to compare effectively HNNM with other algorithms, we select adaptive merging and growing algorithm (AMGA), basic constructive algorithm (BCA), basic constructive–pruning algorithm (BCPA) and Ludermir algorithm[14] as candidates, also, we will take same experimental methodologies as possible.

Table 1 shows the results of HNNM, AMGA,BCA,BCPA and Ludermir over 50 independent runs for Diabetes data set. It can be seen that HNNM has the smallest number of hidden neurons and number of epochs, while BCP is largest in term of number of hidden neurons. With respect to TER, HNNM takes the lowest.

Table 1. The result of testing Diabetes Data set

Algorithm	Number of hidden neurons	TER	Number of epochs
HNNM	5.05	21.55	420
AMGA	4.14	21.97	390
BCA	5.96	26.04	467
BCPA	5.56	26.22	501
Ludermir	4.53	25.87	--

5.3 Discussion

From the experimental results, HNNM takes better performance than other algorithms in terms of TER. There are three reasons. One is that the neural networks are trained by GA, which can help to find a global best solution, and avoid procedure being trapped into a local minimum. Next is that HNNM uses an adaptive strategy like AMGA, and can dynamically prune or add the hidden neurons, while, other algorithms use a fixed modification strategy. Last is that training parameters is taken by RL agent. Unlike AMGA, in HNNM, RL agent evaluates significance of hidden neurons according to previous training results, feedback of current training and expert's knowledge. It isn't necessary to compute the significance according to empirical formula. HNNM is easy to be suitable for all classification problems.

6 Conclusion

This paper has present a hybrid neural network learning model based on reinforcement learning agents. In this model, NN agent is responsible of training the neural networks by using GA algorithm, while RL agent is responsible of determining the parameters of the neural networks by reinforcement learning algorithm, such as significance of hidden neurons, epochs, errors, initial weights, and hidden neurons, etc. There are two main contributions in this paper. First, due to being trained by GA, it's easy that HNNM can find a better global solution. Second, significance of the hidden neurons that will be merged may not be computed by formula, but determined by RL agent according to the evaluation at their influences on training results (through receiving reward and observing state of ANN), so that HNNM has the ability to generalize the training algorithm of ANN. A series of simulations are provided to demonstrate these.

References

1. Tsai, J., Chou, J., Liu, T.: Tuning the structure and parameters of a neural network by using hybrid Taguchi-genetic algorithm. IEEE Trans. Neural Netw. 17, 69–80 (2006)
2. Teoh, E.J., Tan, K.C., Xiang, C.: Estimating the number of hidden neurons in a feedforward network using the singular value decomposition. IEEE Trans. Neural Netw. 17, 1623–1629 (2006)
3. Islam, M., Sattar, A., Amin, F., Yao, X., Murase, K.: A New Constructive Algorithm for Architectural and Functional Adaptation of Artificial Neural Networks. IEEE Trans. on Systems, Man and Cybernetics—Part B: Cybernetics 39, 1590–1605 (2009)

4. Goh, C.K., Teoh, E.J., Tan, K.C.: Hybrid Multiobjective Evolutionary Design for Artificial Neural Networks. IEEE Trans. Neural Netw. 19, 1531–1547 (2008)
5. Hamid, B., Mohamad, R.M.: A learning automata-based algorithm for determination of the number of hidden units for three-layer neural networks. International Journal of Systems Science 40, 101–118 (2009)
6. Islam, M., Sattar, A., Amin, F., Yao, X., Murase, K.: A New Adaptive Merging and Growing Algorithm for Designing Artificial Neural Networks. IEEE Trans. on Systems, Man and Cybernetics—Part B: Cybernetics 39, 705–718 (2009)
7. Gao, P.Y., Chen, C.B., Qin, S., Hu, Y.S.: An Optimization Method for Neural Network Based on GA and TS Algorithm. In: 2nd International Conference on Computer and Automation Engineering. IEEE Press, New York (2010)
8. Farhang, S., Hamid, R.T., Magdy, M.M.A.S.: A reinforcement agent for object segmentation in ultrasound images. Expert Systems with Applications 35, 772–780 (2008)
9. Ronnie, W., Robert, W.: 2009 Special Issue: Representation in dynamical agents. Neural Networks 22, 258–266 (2009)
10. Tan, A.H.: Self-organizing neural architecture for reinforcement learning. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 470–475. Springer, Heidelberg (2006)
11. Benardos, P.G., Vosniakos, G.C.: Optimizing feedforward artificial neural network architecture. Engineering Application of Artificial Intelligence 20, 365–382 (2007)
12. Zhang, K., Andrew, B., Gu, F.S., Yu, H., Li, S.: A hybrid model with a weighted voting scheme for feature selection in machinery condition monitoring. In: Proc. of 3rd IEEE International Conference on Automation Science and Engineering, pp. 424–429. IEEE Press, New York (2007)
13. Sasakawa, T., Hu, J.L., Hirasawa, K.: A Brainlike Learning System with Supervised, Unsupervised, and Reinforcement Learning. Electrical Engineering in Japan 162, 32–38 (2008)
14. Ludermir, T.B., Akio, Y., Cleber, Z.: An optimization methodology for neural network weights and architectures. IEEE Trans. Neural Netw. 17, 1452–1457 (2006)

A Multi-view Regularization Method for Semi-supervised Learning

Jiao Wang, Siwei Luo, and Yan Li

School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China

Abstract. Multi-view semi-supervised learning is a hot research topic recently. In this paper, we consider the regularization problem in multi-view semi-supervised learning. A regularization method adaptive to the given data is proposed, which can use unlabeled data to adjust the degree of regularization automatically. This new regularization method comprises two levels of regularization simultaneously. Experimental evidence on real word dataset shows its effectivity.

Keywords: multi-view learning, semi-supervised learning, regularization, machine learning.

1 Introduction

Semi-supervised learning explores how unlabeled data can be used to improve performance of traditional supervised learning [1]. Recently, the multi-view paradigm for semi-supervised learning has received increasing attention [2]. In the multi-view setting, the input variable x can be described with two (or more) different views, a typical example is to classify web pages, each of which can be represented by either the words on itself, or the words contained in the hyperlinks.

It is shown that multi-view learning can get better performance than the single view counterpart [3], especially when the strengths of one view complement the weaknesses of the other. Approaches to multi-view learning can be grouped into two categories. In the first category, a set of classifiers defined in each view are mutually trained in an iterative process [2, 4]. In the second category, unlabeled data are used to find a better feature representation, so that it is easy to learn with labeled data in the new feature space [5, 6].

Many practical semi-supervised learning methods face the problem of overfit. In this paper, we propose a multi-view regularization approach to semi-supervised learning. There are two levels of regularization in our approach, within-view regularization and between-view regularization. In the within-view regularization, the unlabeled data are used to adjust the degree of regularization automatically. In the between-view regularization, the algorithms try to maximize the consensus of each view on the unlabeled data, which reduce the complexity of the learning problem by eliminate hypothesis that disagree with each other on the unlabeled data.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 444–449, 2010.

2 Multi-view Regularization for Semi-supervised Learning

In semi-supervised learning, there are not only L labeled pairs $(x_1, y_1),...,(x_L, y_L)$, but also U unlabeled instances $x_{L+1},...,x_{L+U}$. Let $x_k = (x_k^1,...,x_k^V)$ be a multi-view sample with V views, where each view can be seen as a set of features. Let $f^v : x^v \to y$ be the classifier that we seek in each view, where $v \in (1,...,V)$.

2.1 Within-View Regularization

In each view of the data, we want to learn a function or hypothesis which effectively predicts the y given any x^v. If a hypothesis is too complex for the data, it may be over-fitting, that is, it will have large test error while the training error is small. Regularization methods figure out this problem by penalizing the global smoothness property of the hypothesis. Usually they solve the following minimization problem:

$$f^* = \arg\min_{f \in H_K} \frac{1}{L} \sum_{i=1}^{L} err(x_i, y_i, f) + \gamma \left\| f \right\|_K^2 \tag{1}$$

where H_K is an Reproducing Kernel Hilbert space of functions with kernel function K; and err is some loss function, such as squared loss for Regularized Least Squares or the hinge loss faction for Support Vector Machines. These methods have shown impressive improvements over naïve supervised learning. However, one difficulty with these techniques is that they usually require expertise to set proper parameters or predefine a norm.

In this paper, we use a data-specific regularization method to penalize model complexity [7], which is parameter-free. In semi-supervised learning, instead of minimizing training error on labeled data alone, we search for hypotheses that behave similarly on labeled and unlabeled data. The intuition is that if the behavior of a hypothesis on labeled data doesn't consistent with it's behavior on unlabeled data, it is not likely to generalize to unseen examples.

Since the exact label of unlabeled data is not known, the behavior of a hypothesis on unlabeled data can not be measured directly. However, the difference between two hypotheses can be obtained using unlabeled data. In order to make use of the difference between two hypotheses to seek a good hypothesis f^v, we can construct a fixed smooth origin function h, e.g. $h = \overline{y}$, and estimate the difference of f^v and h on both labeled and unlabeled data sets, represented by $\hat{d}(f^v, h)$ and $\tilde{d}(f^v, h)$ respectively. Note that we use $\hat{d}$ to denote the difference on labeled data, and $\tilde{d}$ to denote the difference on unlabeled data. Our criterion is to choose the hypotheses that behaves similarly on labeled and unlabeled data, i.e. the hypotheses that makes $\hat{d}(f^v, h)$ equal $\tilde{d}(f^v, h)$ as much as possible, which can be achived by making

$\frac{\tilde{d}(f^v,h)}{\hat{d}(f^v,h)}$ equal 1 as much as possible. Let $\hat{d}(f^v,y)$ represent the difference of f^v and the true label y on labeled data, which means the empirical error. Our objective is to minimize the following equation:

$$\min \hat{d}(f^v,y)\times\frac{\tilde{d}(f^v,h)}{\hat{d}(f^v,h)} \tag{2}$$

Since $\tilde{d}(f^v,h)\geq\hat{d}(f^v,h)$ in general, the following equation holds:

$$\frac{\tilde{d}(f^v,h)}{\hat{d}(f^v,h)}\geq 1 \tag{3}$$

When equation (3) takes "=", equation (2) will be minimized.

To avoid the final hypotheses to be biased towards the origin function h, we use symmetric forms of the above objective that also penalize hypotheses close to h:

$$\hat{d}(f^v,y)\times\max\left(\frac{\tilde{d}(f^v,h)}{\hat{d}(f^v,h)},\frac{\hat{d}(f^v,h)}{\tilde{d}(f^v,h)}\right) \tag{4}$$

which is called the within-view loss function L_V, the term $\tilde{d}(f^v,h)$ uses unlabeled data to automatically set the level of regularization, it penalizes the smoothness property of the hypothesis in a parameter-free way.

2.2 Between-View Regularization

When using the hypotheses got in each view to classify the unlabeled data, there is much chance that they disagree with each other. Intuitively, for a certain instance, if the predictions of these hypotheses are consistent with each other, the predictions have more probability to be true, as these hypotheses are not likely to make mistake simultaneously.

Dasgupta et al. [8] studied the relation between the consensus of two independent hypotheses and their error rate, one of their results that hold if the error rate of either hypothesis is smaller than 1/2 is the inequality:

$$P\left(f^1\neq f^2\right)\geq\max_{v=1,2}P\left(f^v\neq y\right) \tag{5}$$

That is to say, the probability of a disagreement between two independent hypotheses upper bounds the error rate of either hypothesis. In fact, even if the two hypotheses are not independent, the disagreement between them is also a good way to upper bound the error rate of either hypothesis. In this way, to minimize the error rate of

hypotheses, we can minimize their disagreement on unlabeled data, which is equal to minimize the difference between any two hypotheses on unlabeled data:

$$\min \tilde{d}(f^{v_1}, f^{v_2}) \tag{6}$$

which is called the between-view loss function L_B, where $v_1, v_2 \in (1,...,V)$. In this way, we use the unlabeled data to adjust the degree of between-view regularization automatically.

2.3 Combining Two Levels of Regularization

To seek the consistent hypotheses between views and keep them smooth in each view, we combine two levels of regularization together:

$$\sum_{v=1}^{V} L_v + \lambda \sum_{v_1, v_2=1}^{V} L_B \tag{7}$$

which has to be minimized. The scalar λ determines the influence of each regularization. The first term measures the within-view loss, the last term measures the between-view loss, both in a data-dependent fashion. That is, the regularization is adaptive to the observed data. This raises the possibility of outperforming other regularization methods that keep fixed across different observed data.

We estimate the difference between two hypotheses on unlabeled data and labeled data by:

$$\tilde{d}(f,g) \triangleq \varphi\left(\frac{1}{U}\sum_{i=1}^{U} err(f(x_i), g(x_i))\right) \tag{8}$$

$$\hat{d}(f,g) \triangleq \varphi\left(\frac{1}{L}\sum_{i=1}^{L} err(f(x_i), g(x_i))\right) \tag{9}$$

For regression problem, err can be squared loss, that is

$$err(f,g) = (f-g)^2 \tag{10}$$

for classification problem, err can be misclassification loss, that is

$$err(f,g) = 1_{(f \neq g)} \tag{11}$$

φ is a function that ensure d to satisfy metric properties, that is, non-negativity, symmetry, and the triangle inequality. Most functions admit a metric, for example, for regression problem, $\varphi(z) = z^{1/2}$; for classification problem, $\varphi(z) = z$.

The minimization objective in Equation (6) is not convex. We set several initialization, and choose the best result. For given initial values, the local optimization solution can be obtained by any nonlinear optimization technique.

3 Experimental Results

To evaluate the proposed regularization algorithm, in this section we compare it with the state-of-the-art related methods on real-world classification problems. The methods we compare with are co-training [2], co-regularization [9], and regularized least square (RLS) [10]. Co-training is the original work in multi-view semi-supervised learning. Co-regularization is a popular multi-view regularization algorithm. RLS is the classical regularization method in machine learning.

We perform experiments on the WebKB dataset which deal with hypertext categorization task. We choose this dataset because it is studied in [2, 9] and it is common used in multi-view semi-supervised. The WebKB database contains 1051 web pages, collected from the websites of computer science faculties of four different universities. For each web page, the database contains both the words contained in the page itself (referred to as View 1) and words appearing in links pointing to that web pages (View 2). The data was preprocessed into 3000 features for View 1 and 1840 features for View 2 using the Rainbow software [11]. The web pages are split into two classes, course and non-course pages. The goal of the learning algorithms presented in this section is to correctly classify web pages into these two classes.

For each method, 10-fold cross-validation is used to get a mean result. The final output of the multi-view learning methods is computed as $f^* = \frac{1}{V}\sum_{v=1}^{V} f^v$. The parameter λ in Equation (6) is set to 1 for the experiments. The iteration number of co-training is set to 100, and the base learner of co-training is naïve bayes. The parameter of co-regularization is set as in [9].

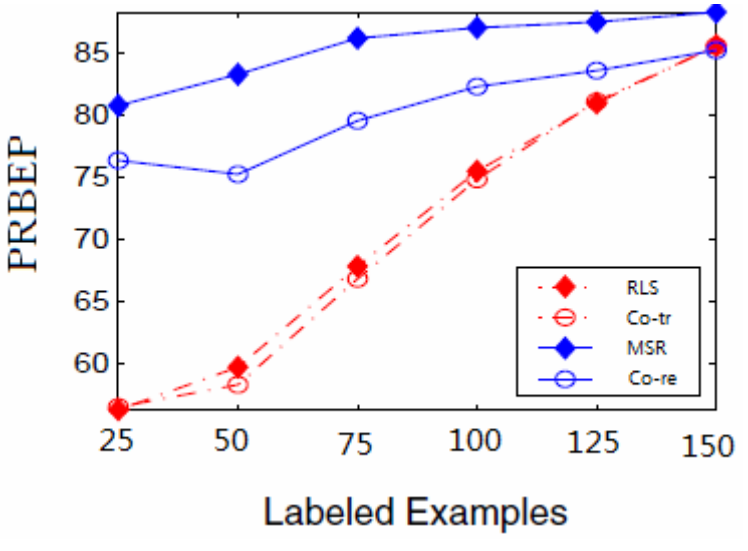


Fig. 1. The precision-recall breakeven points measured as the function of the number of labeled data

Figure 1 measures the precision-recall breakeven points of these methods, where MSR represents Multi-view Semi-supervised Regularization proposed by us. It can be seen from figure 1 that MSR outperform other methods distinctly.

4 Conclusion

Regularization is a useful method in machine learning to find a proper function from the function space. In this paper, we propose a multi-view semi-supervised regularization algorithm, it use unlabeled data to adjust the degree of regularization automatically. Experiments results illustrate that the proposed regularization algorithm outperform other related works in this field.

In future works, we will analyze the generation error bound of the proposed algorithm, and study the complexity of this algorithm.

Acknowledgements. This work is supported by National High Technology Research and Development Program of China (2007AA01Z168), National Nature Science Foundation of China (60773016, 60805041, 60872082, 60975078, 60902058), Beijing Natural Science Foundation (4092033), and Doctoral Foundations of Ministry of Education of China (200800041049).

References

[1] Chapelle, O., Scholkopf, B., Zien, A.: Semi-Supervised Learning. MIT Press, Cambridge (2006)
[2] Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the 11th Annual Conference on Computational Learning Theory, Madison, WI, pp. 92–100 (1998)
[3] Kakade, S.M., Foster, D.P.: Multi-view regression via canonical correlation analysis. In: The 20th Annual Conference on Learning Theory, pp. 82–96 (2007)
[4] Zhou, Z.-H., Li, M.: Tri-training: exploiting unlabeled data using three classifiers. IEEE Transactions on knowledge and data engineering 17(11), 1529–1541 (2005)
[5] Zhou, Z.-H., Zhan, D.-C., Yang, Q.: Semi-supervised learning with very few labeled training examples. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence, Vancouver, Canada, pp. 675–680 (2007)
[6] Peng, Y., Zhang, D.Q.: Semi-supervised canonical correlation analysis. Journal of Software 19(11), 2822–2832 (2008) (in chinese)
[7] Schuurmans, D., Southey, F.: Metric-based methods for adaptive model selection and regularization. Machine Learning 48(1-3), 51–84 (2002)
[8] Dasgupta, S., Littman, M.L., McAllester, D.: PAC generalization bounds for co-training. In: Advances in Neural Information Processing Systems, vol. 14, pp. 375–382. MIT Press, Cambridge (2002)
[9] Sindhwani, V., Niyogi, P., Belkin, M.: A co-regularized approach to semi-supervised learning with multiple views. In: Working Notes of the ICML 2005 Workshop on Learning with Multiple Views, Bonn, Germany (2005)
[10] Vapnik, V.N.: Statistical Learning Theory. Wiley Interscience, New York (1998)
[11] McCallum, A.: Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering (1996), `http://www.cs.cmu.edu/»mccallum/bow`

Multi-reservoir Echo State Network with Sparse Bayesian Learning

Min Han and Dayun Mu

School of Electronic and Information Engineering, Dalian University of Technology, Dalian, Liaoning 116023, China
minhan@dlut.edu.cn

Abstract. A multi-reservoir Echo State Network based on the Sparse Bayesian method (MrBESN) is proposed in this paper. When multivariate time series are predicted with single reservoir ESN model, the dimensions of phase-space reconstruction can be only selected a single value, which can not portray respectively the dynamic feature of complex system. To some extent, that limits the freedom degree of the prediction model and has bad effect on the predicted result. MrBESN will expand the simple input into high-dimesional feature vector and provide the automatic estimation of the hyper-parameters with Sparse Bayesian. A simulation example, that is a set of real world time series, is used to demonstrate the validity of the proposed method.

Keywords: Multi-reservoir; ESN; Sparse Bayesian; Time series prediction.

1 Introduction

Neural Networks (NN) has played a crucial role in time series prediction. The Recurrent Neural Network (RNN) is a NN-based model with consideration of the internal feed-back, which has proved to be an efficient algorithm for time series prediction and can overcome the inherent limitations of the feed-forward neural network [1]. Echo State network (ESN) is a novel recurrent neural network. The ESN approach differs from traditional RNN methods in that a large RNN is used (order of 50 to 1000 neurons, previous techniques typically use 5 to 30 neurons) and only the synaptic connections from the RNN to the output readout neurons are modified by learning (previous techniques tune all synaptic connections) [2]. Training ESN becomes a simple linear regression task, which solves the problems of slow convergence and suboptimal solutions in previous methods. Recently, the support vector echo-state machine (SVESM) is proposed for predicting the chaotic time series [3]. The basic idea is to replace the "kernel trick" with "reservoir trick" in nonlinear system modeling, that is to say, performing linear SVR in the high-dimension "reservoir" state space.

However, the complex system usually comprises many variables, and when multivariate time series are predicted with single reservoir ESN model [2, 3], the dimensions of phase-space reconstruction can be only selected a single value, which can not portray respectively the dynamic feature of complex system. To some extent, that limits the freedom degree of the prediction model and has bad effect on the predicted

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 450–456, 2010.

accuracy. Meanwhile, there are some "nuisance" parameters introduced by SVESM which were determined in an invalid method. The cross validation is a common method used to solve this problem, but it is very time-consuming to find appropriate parameters.

In this paper, the Multi-reservoir ESN based on Sparse Bayesian method is proposed to solve the above problems, so that the prediction can be of high accuracy. Every variable of the complex system can be adequately mapped into characteristic space by way of every reservoir so that the dynamic characteristic of the complex system is fully portrayed. Meanwhile, Sparse Bayesian method is applied to calculate the output weight which reduces the complicacy of prediction model and overcomes the over-fitting. Consequently the accuracy of multivariate time series prediction is enhanced obviously.

The remaining paper is organized as follows. Section 2 introduces ESN and presents the architecture of Multi-reservoir Echo State Network. Section 3 presents the improved Bayesian Method for Multi-reservoir ESN (MrBESN). Section 4, a real-world time series which are the annual runoff of Yellow River and sunspots time series is applied to test the proposed model. Discussions and conclusions are given in Section 5.

2 The Architecture and Training Algorithm of Multi-reservoir ESN

2.1 Short Introduction to Echo State Network

Echo state network was first proposed by Jaeger and Haas to learn nonlinear systems and predict chaotic time series. Its basic idea is to use a large "reservoir" RNN.

In the ESN approach, a larger-than-normal layer of neurons is used with random recurrent connections where the numeric value ranges from -1 to 1, which is not modified during training. They serve as a 'dynamic reservoir' (DR) [2]. The network has an output layer, and may also have an input layer. Input-to-hidden layer connections are also randomly generated from -1 to 1 and are not modified during training. The hidden-to-output connections W_{out} are trained as Eq.1:

$$(W^{out})' = M^{-1}T \tag{1}$$

Where M is the whole internal state matrix through sample period and M^{-1} is the pseudoinverse of M, T is the teacher signal vector.

2.2 Architecture and Training Algorithm of Multi-reservoir ESN

The complex system usually comprises many variables, and when multivariate time series are predicted with single reservoir ESN model, the dimensions of phase-space reconstruction can be only selected a single value, which can not portray respectively the dynamic feature of complex system. To some extent, that limits the freedom degree of the prediction model and has bad effect on the predicted accuracy. To solve the above problem, Multi-reservoir ESN is presented in this paper. Every variable of the complex system can be adequately mapped into characteristic space by way of

every reservoir so that the dynamic speciality of the complex system is fully portrayed. The architecture of Multi-reservoir ESN is as fig.1:

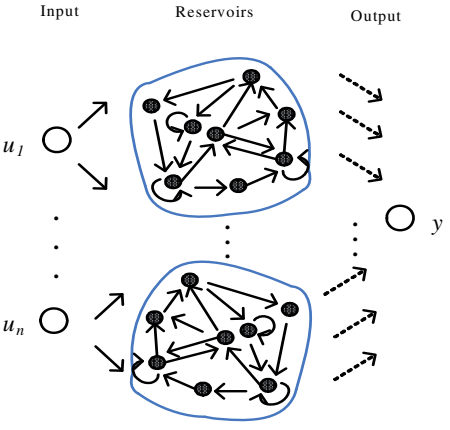


Fig. 1. Structure of Multi-reservoir ESN

And the equation of Multi-reservoir ESN (take two reservoirs for example) can be then represented as

$$\begin{aligned} x_1(k+1) &= \tanh(W_x \cdot x_1(k) + W_{in} \cdot u_1(k)) \\ x_2(k+1) &= \tanh(W_x \cdot x_2(k) + W_{in} \cdot u_2(k)) \end{aligned} \tag{2}$$

$$y(k+1) = w_1^T x_1(k) + w_2^T x_2(k) + \varepsilon \tag{3}$$

where W_{in} represents a weight matrix from the input neurons to the internal ones; W_x represents the internal connection matrix; $u_1(k)$ and $u_2(k)$ are the input fed into the network at time step k ; $x_1(k)$ and $x_2(k)$ are the internal state matrix at time step k; $y(k+1)$ is the output at time step $k+1$.The output of the network is computed according to the linear function as the equation (3), where w_1 and w_2 represent connections to the output units, and ε is noisy signal. One of the key features of ESN is that only the weights of the connections to the output units w_1, w_2should be modified during learning/training process. The topology of the hidden layer and other weight matrixes remain unchanged. Because there are no cyclic dependencies between the trained readout connections, any linear regression method is available. Therefore, we apply the Sparse Bayesian regression algorithm to learn the W_{out} during training.

3 Sparse Bayesian Method for Multi-reservoir ESN

3.1 The Prediction Model

The equations of the Bayesian Multi-reservoir ESN in Eq.(2),(3)can be represented as the following form:

$$
\begin{aligned}
x_1(k+1) &= \tanh(W_x \cdot x_1(k) + W_{in} \cdot u_1(k)) \\
x_2(k+1) &= \tanh(W_x \cdot x_2(k) + W_{in} \cdot u_2(k)) \\
\mathrm{Y} = y(k+1) &= w_1^T x_1(k) + w_2^T x_2(k) + \varepsilon \\
&= \begin{bmatrix} w_1^T & w_2^T \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \varepsilon \\
&= \mathbf{W}^T \mathbf{X} + \varepsilon
\end{aligned} \tag{4}
$$

Where tanh denotes hyper-bolic tangent sigmoid function which is applied. Because there are no cyclic dependencies between the trained output weights, training Multi-reservoir ESN becomes a linear regression task. We use the Sparse Bayesian linear regression to solve this problem.

$\boldsymbol{\varepsilon}$ is a zero mean Gaussian random variable with precision β. Due to the assumption of independent of $y(k)$, the likelihood of the complete data set can be written as

$$
p(\mathbf{Y} \mid \mathbf{w}, \beta) = \prod_{k=1}^{\hat{N}} \mathrm{N}(y(k) \mid \mathbf{w}^T \mathbf{x}(k), \beta^{-1}) \tag{5}
$$

For clarity, we omit to notate the implicit conditioning upon the set of the state matrix $\mathbf{x}$ and subsequent expression. The Equation (5) is the exponential of a quadratic function of $\mathbf{W}$.

With as many parameters in the model as training examples, we would expect maximum likelihood estimation of $\mathbf{w}$ and β from (5) to lead to severe over-fitting. To avoid this, a common approach is to impose some additional constraint on the parameters, for example, through the addition of a 'complexity' penalty term to the likelihood or error function. Here, we adopt a Bayesian perspective, and 'constrain' the parameters by defining an explicit prior probability distribution over them[4].

We encode a preference for smoother (less complex) functions by making the popular choice of a zero-mean Gaussian prior distribution over $\mathbf{w}$:

$$
p(\mathbf{w}) = \mathrm{N}(\mathbf{w} \mid \mathbf{m}_0, \mathbf{S}_0) \tag{6}
$$

In order to simplify the treatment, we consider a zero-mean isotropic Gaussian governed by a single precision parameter α so that $\mathbf{m}_0=0$ and $\mathbf{S}_0=\alpha^{-1}\mathrm{I}$.

To compute the posterior distribution, which is proportional to the product of the likelihood function and the prior, we use Bayesian theorem as equation (7).

$$
p(\mathbf{w} \mid \mathbf{Y}) = \frac{p(\mathbf{Y} \mid \mathbf{w}) P(\mathbf{w})}{p(\mathbf{Y})} \propto p(\mathbf{Y} \mid \mathbf{w}) P(\mathbf{w}) \tag{7}
$$

Due to the choice of a conjugate Gaussian prior distribution, the posterior will also be Gaussian. We can evaluate this distribution by the usual procedure of completing the square in the exponential, and then finding the normalization coefficient using the standard result for a normalized Gaussian [5]. The prior distribution is shown as

$$
p(\mathbf{w} \mid \mathbf{Y}) = \mathrm{N}(\mathbf{w} \mid \mathbf{m}_N, S_N) \tag{8}
$$

Where

$$\mathbf{m}_N = \beta S_N \mathrm{X}^{\mathrm{T}} y,\ S_N^{-1} = \alpha \mathbf{I} + \beta \mathrm{X}^{\mathrm{T}} \mathrm{X} \tag{9}$$

The log of the posterior distribution is given by the sum of the log likelihood and the log of the prior and, as a function of **w**, takes the form

$$\ln p(\mathbf{w}|\mathbf{Y}) = -\frac{\beta}{2}\sum_{k=1}^{\hat{N}}\left\{ y(k+1) - \mathbf{w}^T \mathbf{x}(k)\right\}^2 - \frac{\alpha}{2}\mathbf{w}^T\mathbf{w} + \mathbf{const} \tag{10}$$

From the following section, we know that the Sparse Bayesian ESN becomes the search of the hyper-parameter posterior mode, Values of α and β can be obtained evidence procedure[6,7].

3.2 Evaluation of the Prediction

Prediction is to build a model to approximate the unknown nonlinear function mapping of the observed signal. To quantitatively measure the performance of proposed prediction model, root-mean-square error (rmse) will be used as indicators in following simulation

$$E_{\mathrm{rmse}} = \left(\frac{1}{N-1}\sum_{k=1}^{N}\left[\hat{y}(k) - y(k)\right]^2 \right)^{1/2} \tag{10}$$

Where $\hat{y}(k)$ is the target value and $y(k)$ is the predicted value, E_{rmse} can be used to describe how well the model accounts for the variation in the observed data. In ideal situation, if there were no error in prediction, these parameters would indicate like the situation that $E_{rmse} = 0$.

4 Experiments and Results

MrBESN is used to estimate the series of annual mean sunspots and natural annual runoff of Yellow River measured at Sanmenxia gauge station from 1700 to 2003 about 304 years. The ESN and SVESM are used as the comparison to the proposed MrBESN. The parameters of the ESN and SVESM are the same as the MrBESN in Table 1.

Table 1. Parameters settings of MrBESN

Item	Value
Size of the Reservoir	100×100
Sparseness of *Wx*	5%
Spectral Radius of *Wx*	1
Input weight connections	Uniform from[-1 1]

According to the correlation dimension method and pseudo-nearest neighbor method, the phase-space reconstruction parameters of the sunspots and the Yellow River runoff time series are set as followed: Delay time are 4 and 1 and Embedding dimension are 2 and 6 respectively. Reconstructed multi-variables are used as the input signal of the Multi-reservoir prediction model. We use the former 260 sets of datas for training model and the remaining datas for prediction. The corresponding forecast results is as shown in Table 2.

Table 2. Prediction results from different models

Model	MrBESN	SVESM[3]	ESN[2]
E_{rmse}(sunspots)	13.665	15.404	23.745
E_{rmse}(runoff)	43.7003	50.720	48.569

Table 2 shows that the prediction model based on method presented in this paper is more accurate and effective.Fig. 2 and Fig. 3 show the performance of the annual sunspots and runoff of the Yellow River time series prediction with MrBESN.

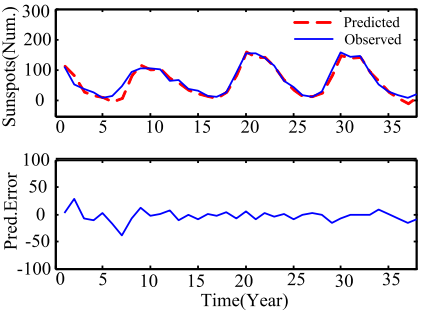


Fig. 2. Comparison between the predicted and observed curves of sunspot time series and their errors

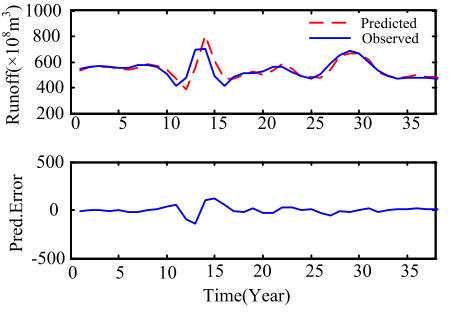


Fig. 3. Comparison between the predicted and observed curves of annual runoff time series of Yellow River and their errors

4 Conclusion

In this paper, we proposed a multi-reservoir Echo State Network which is based on Sparse Bayesian method (MrBESN), gave the Sparse Bayesian regression method and the method to estimate the hyper-parameters. Its performance has been tested by an example which are a real-world time series. They are the annual runoff of Yellow River and sunspots time series. The simulation results shows that the prediction model based on method presented in this paper is more accurate and effective.

Acknowledgements

This research is supported by the project (60674073) of the National Nature Science Foundation of China, the project (2007AA04Z158) of the National High Technology Research and Development Program of China (863 Program), the project (2006B AB14B05) of the National Key Technology R&D Program of China and the project (2006CB403405) of the National Basic Research Program of China (973 Program).

References

1. Maier, H.R., Dandy, G.C.: Neural networks for the prediction and forecasting of water resources variables: A review of modeling issues and applications. Environmental Modelling & Software 15, 101–124 (2000)
2. Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science 304(5667), 78–80 (2004)
3. Shi, Z., Han, M.: Support Vector Echo-State Machine for Chaotic Time-Series Prediction. IEEE Transactions on Neural Networks 18(2), 359–372 (2007)
4. MacKay, D.J.C.: Bayesian interpolation. Neural Computation 4(3), 415–447 (1992)
5. Ming, Z.M., Tao, J.Y.: Bayesian demonstration test method with mixed beta distribution. Chinese Journal of Mechanical Engineering (English Edition) 3(21), 116–119 (2008)
6. Tipping, M.E.: Sparse Bayesian Learning and the Relevance Vector Machine. Journal of Machine Learning Research 1, 211–244 (2001)
7. MacKay, D.J.C.: Comparison of approximate methods for handling hyperparameters. Neural Computation 11(5), 1035–1068 (1999)

Leave-One-Out Cross-Validation Based Model Selection for Manifold Regularization

Jin Yuan[1], Yan-Ming Li[1], Cheng-Liang Liu[1], and Xuan F. Zha[2]

[1] School of Mechanical Engineering, Shanghai Jiao Tong University, China
[2] National Institute of Standards and Technology, USA
jinyuan@sjtu.edu.cn

Abstract. Classified labels are expensive by virtue of the utilization of field knowledge while the unlabeled data contains significant information, which can not be explored by supervised learning. The Manifold Regularization (MR) based semi-supervised learning (SSL) could explores information from both labeled and unlabeled data. Moreover, the model selection of MR seriously affects its predictive performance due to the inherent additional geometry regularizer of SSL. In this paper, a leave-one-out cross-validation based PRESS criterion is first presented for model selection of MR to choose appropriate regularization coefficients and kernel parameters. The Manifold regularization and model selection algorithm are employed to a real-life benchmark dataset. The proposed approach, leveraged by effectively exploiting the embedded intrinsic geometric manifolds, outperforms the original MR and supervised learning approaches.

Keywords: Semi-Supervised Learning (SSL); Leave-one-out cross-validation (LOOCV); Manifold Regularization (MR); Model selection.

1 Introduction

Many developed methods in machine learning and data mining promote the practicality of real-life applications such as condition monitoring based fault detection. Classified patterns are fairly expensive to obtain because they require human effort by virtue of the utilization of field knowledge, but unlabeled training data are readily available. Recently, Semi-Supervised Learning (SSL) [1] has attracted attention because of its utilization not only from the labeled training data, but also the structural information in easily available unlabeled data.

Based on the underlying assumptions, Many semi-supervised learning algorithms studied in the literature fall into five categories: SSL with generative models, SSL with low density separation, graph-based methods, co-training methods, and self-training methods [1]. The graph-based methods with more applicable assumption have attracted considerable attention. Specifically, graph-based manifold regularization [2] exploits the geometric structure of the marginal distribution of the data in the feature space. The incorporation of unlabeled data has demonstrated the potential for improved accuracy in time series prediction [3], in

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 457–464, 2010.

speech recognition [4], calibration-effort reduction problem [5], Visual Mapping [6] and etc.

However, the generalization performance of the manifold regularization approach is heavily dependent on the model selection process, in this case the well selection of an appropriate values for the regularization coefficients and kernel parameters is very critical for real-life applications. This paper is concerned with model selection of manifold regularization based on minimization of the leave-one-out cross-validation criteria of the model selection, which can be performed very efficiently for the manifold regularization of kernel learning methods.

2 Manifold Regularization Based Semi-supervised Learning

2.1 Manifold and Cluster Assumptions

A basic assumption often used in machine learning is that nearby points are likely to have the same label. That is, pairwise similarity measure is the basis of label propagation. Although very useful and widely applicable, this smooth assumption sometimes is too weak. Most real-world dataset (like the Speech Sounds [4]) has more complex structures than this assumption could arise, such as the samples clustered in a certain region may have the same labels. In addition, the manifold learning approach [7] extracted the intrinsic manifold features from high-dimensional fault data by directly learning the data and translates complex mode space into a low-dimensional feature space.

So, to take advantage of unlabeled data, manifold regularization introduced another two basic assumptions or priors about the data distribution. The first one is called the manifold assumption, which assumes the data points as forming a low-dimensional manifold in some feature space. The manifold regularization use the graph Laplacian of a graph-based representation to characterize the manifold structure [2]. The second one, cluster assumption [8] is : if two patterns in feature space of samples, e.g. extracted from vibration signal in same condition, are close in the embedded manifold of the marginal distribution. And their conditional distributions are similar to each other, which favors decision boundaries for classification passing through low-density regions in the input space [9].

2.2 Manifold Regularization Framework

This Manifold regularization combines the ideas of spectral graph theory, manifold learning and kernel methods in a coherent and natural way to incorporate both the cluster assumption and the manifold assumption in Reproducing Kernel Hilbert Spaces (RKHS) regularization framework. In this section, we address the manifold regularization based SSL framework concisely followed the description of [2]. More details refer to [10].

Consider a set of l labeled samples $\{(x_i, z_i)\}_{i=1}^{l}$ and a set of u unlabeled samples $\{x_j\}_{j=l+1}^{l+u}$, where $x_i, x_j \in \mathbb{R}^d$ are the feature vectors collected from a

condition monitoring system and $z_i \in \mathbb{R}$ is the classified label. According to the marginal distribution $\mathcal{P}_{\mathcal{X}}$ and z_i are governed by the conditional distribution $\mathcal{P}(z|x)$. Manifold regularization introduces to the regularized risk functional an additional regularizer that serves to impose this assumption on the learning problem. The learning problem corresponds to solving the following optimization problem:

$$f^* = \arg\min_{f \in \mathrm{H}_K} \frac{1}{l} \sum\nolimits_{i=1}^{l} \mathcal{V}(x_i, z_i, f) + \gamma_A \|f\|_K^2 + \gamma_I \int_M \langle \nabla_M, \nabla_M \rangle \tag{1}$$

which finds the optimal function f^* in the RKHS space $\mathcal{H}_K$ of functions $f : \mathcal{X} \to \mathbb{R}$ corresponding to a Mercer kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$.

The first term of the regularized risk functional is defined based on the loss function $\mathcal{V}$ which measures the discrepancy between the predicted value $f(x_i)$ and the actual value z_i. The second term controls the complexity of f in terms of the RKHS norm $\|\cdot\|_K$, with γ_A being the RKHS norm regularization parameter. The third term is specific to manifold regularization and is based on the assumption that the support of $\mathcal{P}_{\mathcal{X}}$ forms a compact submanifold $\mathcal{M}$. It controls the complexity of f in the intrinsic geometry of $\mathcal{P}_{\mathcal{X}}$, with γ_I being the corresponding Manifold regularization parameter. The third term is approximated using the graph Laplacian defined on all $l + u$ labeled and unlabeled examples without using the label information. Hence the optimization problem can be reformulated as:

$$f^* = \arg\min_{f \in \mathrm{H}_K} \frac{1}{l} \sum\nolimits_{i=1}^{l} \mathcal{V}(x_i, z_i, f) + \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(l+u)^2} \hat{f}^T L \hat{f} \tag{2}$$

where $\hat{f} = (f(x_1), \ldots, f(x_{l+u}))'$ and L is the graph Laplacian.

From the extended Representer theorem [2], the optimal function can be expressed in the following form:

$$f^*(x) = \sum_{i=1}^{l+u} \alpha_i K(x_i, x) \tag{3}$$

When lose function $\mathcal{V}$ in Eq.2 is adopt to be the squared loss function $V(x_i, z_i, f) = (z_i - f(x))^2$, the Laplacian Regularized Least Squares (LapRLS) algorithm [2] formulates the optimization problem:

$$f^* = \arg\min_{f \in \mathrm{H}_K} \frac{1}{l} \sum\nolimits_{i=1}^{l} (z_i - f(x))^2 + \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(l+u)^2} \hat{f}^T L \hat{f} \tag{4}$$

For the LapRLS, the optimal solution in Eq.3 $\alpha^* = (\alpha_1^*, \ldots, \alpha_{l+u}^*)'$ is given by

$$\alpha^* = (JK + \gamma_A l I + \frac{\gamma_I l}{(l+u)^2} LK)^{-1} Z \tag{5}$$

where K is the $(l+u) \times (l+u)$ Gram matrix over all labeled and unlabeled samples, Z is an $(l+u)$-dimensional label vector given by $Z = (z_1, \ldots, z_l, 0, \ldots, 0)'$,

and $J = (1, \ldots, 1, 0, \ldots, 0)$ is an $(l+u) \times (l+u)$ diagonal matrix with the first l diagonal entries being 1 and the rest being 0.

When lose function $\mathcal{V}$ in Eq.2 is adopt to be the hinge loss function $V(x_i, z_i, f) = (1 - z_i f(x))$, the algorithm formulates the Laplacian Support Vector Machines (LapSVM). The details refer to [2].

3 Model Selection

In machine learning community, it is well known that the model selection plays an essential role in real-life application. A poor kernel parameters in kernel methods will lead to bad performance. How to implement model selection to improve recognition performance is a crucial task. The generalization performance of the Manifold regularization is heavily dependent on the model selection process, in this case the careful selection of appropriate values for the extrinsic and intrinsic regularization and semi-supervised kernel parameters.

3.1 Optimizing Regularization and Kernel Parameters

In the Manifold regularization, some form of model selection is required to determine good values for the hyper-parameters in order to maximize generalization performance. Generally, for the LapSVM and LapRLS, weuse a Gaussian kernel $K(x, z) = e^{-\frac{\|x-z\|^2}{2\sigma^2}}$ and Euclidean nearest neighbor graphs with gaussian weights since it hold consistently across a wide applications. The continuous variable σ is Gaussian kernel width, which has prior weightiness to tune. The number of nearest neighbor $N \in \mathbb{N}^+$ and the degree ($D \in \mathbb{N}^+$) of the graph Laplacian are both discrete hyper-parameters in constructing a nearest neighbor graph and deforming the semi-supervised kernel served as an approximation to the true manifold. We used the inductive setting in the optimization of the extrinsic and intrinsic regularization parameters: γ_A and γ_I to reduce the complexity. The inductive setting in [2] chose a fixed split ratio between $\frac{\gamma_I l}{(l+u)^2}$ and $\gamma_A l$. Here the continuous variable split ratio λ is chosen as optimizing hyper-parameter, while $\gamma_A l$ is fixed to 0.01 because of its insensitive to error rate. So, in this paper, $\theta = (\sigma, \lambda, N, D)$ is selected as optimization variables.

3.2 Model Selection Criteria

Leave-one-out cross-validation has been shown to give an almost unbiased estimator of the generalization properties of statistical models [11]. Empirical studies [12] have shown that in some cases the model selection strategies can be performed very efficiently for kernel learning methods, as well as graph kernel learning of manifold regularization.

In the Manifold regularization, for LapRLS and LapSVM, the analytic leave-one-out cross-validation procedure can easily form the basis of an efficient model selection strategy based on a weighted version of Predicted REsidual Sum of

Squares (PRESS) criterion [13], which is simply the leave-one-out estimate of the sum-of squares error:

$$PRESS(\theta) = \sum_{i=1}^{l} \varsigma_i \left(z_i - \hat{z}_i^{(-i)} \right)^2 \tag{6}$$

In order to tradeoff the contribution of the positive and negative samples misclassification in term of the regularized loss function, a weighted loss function is considered and the weighting factors are chosen according to

$$\varsigma_i = \begin{cases} \frac{\ell}{2\ell^+} \text{ if } z_i = +1 \\ \frac{\ell}{2\ell^-} \text{ otherwise} \end{cases} \tag{7}$$

where ℓ^+ and ℓ^- represent the number of positive and negative examples respectively. Note that the leave-one-out cross-validation procedure is heavily time consumption in middle or large scale problem. This criteria could be converted to minimization of k-fold cross validation, even an independent validation dataset.

3.3 Optimization Trick

The minimization of leave-one-out PRESS criterion with $\hat{z}_i$ computed by Eq.(6) often can be found via gradient based optimization in model selection. Unfortunately, due to the discrete hyper-parameters (N, D), the gradient based optimization like [12] is hard to process, even using the continuous approximation technique [14]. Alternatively, a simple grid-search procedure is adopted in the majority of practical applications [15] of kernel learning methods. However, this is often inefficient as a grid-search spends heavily time on investigating hyper-parameter values if the optimum value is outside the grid.

In this paper, the minimization process is implemented by using a more efficient Nelder-Mead simplex algorithm [16], which is a relatively straight-forward way that does not use numerical or analytic gradients. There are two problems need to deal with in the minimization procedure. First, we use the continuous optimization algorithm and the evaluation with round variables for the continuous and discrete hyper-parameters optimization problem. Second, the nominalized matrix is utilized for mapping unscaled optimized domain to scaled space of the hyper-parameters. The optimized parameters of the simplex optimizer are set by $\delta = 1$ and $\gamma = 0.5$.

4 Empirical Study and Discussion

In this section, we performed two semi-supervised learning experiments on a real world dataset (USPS). To verify the classification and fault detection capability of model selection based manifold regularization, comparisons are made with supervised learning methods and original grid search based manifold regularization approach. The datasets descriptions and parameter setting of the experiment are shown in Table 1.

Table 1. The datasets descriptions and experimental settings

Dataset	Size		Dimensions	classes
	Train (labeled+ unlabeled)	Test		
USPS	1000(50+950)	1007	256	10

4.1 Classification Performance Comparison

Support vector machine (SVM) and Regularized Least Squares (RLS) are state of the art supervised learning methods. From the Table 2 we can see, the semi-supervised learning techniques leveraged by the structure knowledge of combining labeled and unlabeled dataset have better performance in pattern classification on USPS and the various Buma pump datasets. Furthermore, the leave-one-out based model selection of manifold regularization outperforms the grid search based original MR approach. For example, the USPS dataset, LapSVM and LapRLS obtained around 14% improvement over their supervised counterparts and almost 7% further increase upon the semi-supervised model not optimized in regularization and kernel parameters. The model parameters corresponding to LapRLS tuned by model selection are given out in Table 3. Although the improvements of model selection obviously are data dependent, generally, the relative good performance can be achieved using the PRESS criterion based optimization process.

In addition, from Table 2 we also can see, the energy features of WPT coefficient outperform the time-domain features under the same sampling strategy. In particular, the dataset extracted time domain features from high-frequency sampling has bad separability of classes than other dataset. So, it shows that the refined features definition still is first option in fault detection.

Table 2. Classification performance comparison among approaches in test error rates

Dataset	σ	N	D	λ
USPS	2.07	7	2	0.71

Table 3. Results of Model selection of Manifold Regularization for LapRLS approach over one of datasets

Dataset	Supervised learning		Manifold Regularization			
	SVM	RLS	Grid search		Model selection	
			LapRLS	LapSVM	LapRLS	LapSVM
USPS	24.6	24.5	17.6	17.8	9.34	9.76

4.2 Discussion

This experimental results demonstrate that the leave-one-out cross-validation based PRESS criterion provides an effective means of model selection of manifold regularization. However, compared with SVM and other supervised learning

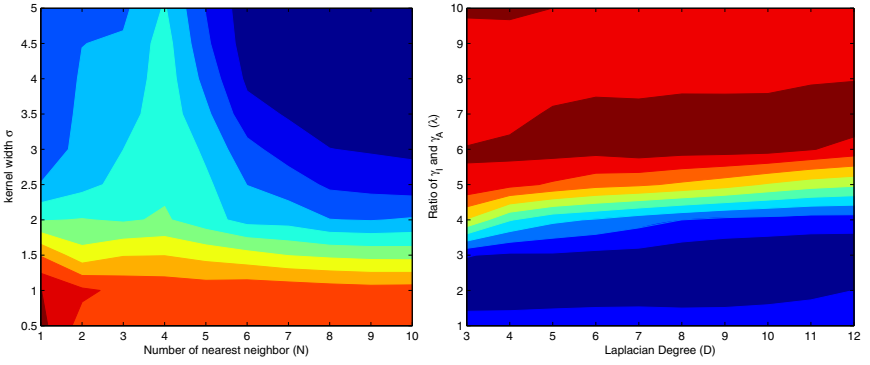


Fig. 1. Local functional characteristics of test error rate vs. pairwise parameters upon USPS dataset classification

techniques, it is note that semi-supervised learning approaches in term of additional introduced regularizer inevitably have more regularization and kernel parameters to be tuned, which bring a little more intractable in the inherent local optimum problem. The contours of test error rate vs. the pairwise parameters in Fig. 1 show the functional complexity of local characteristics. Thus, the Nelder-Mead simplex based optimization processes may still fall into local optimum, in particular for the bad separability dataset, that might means strong nonlinearity decision function. In such situations, the grid search is an external reinforcement tool to avoid selection bias.

5 Conclusions

Manifold regularization combines the hidden structural information in the unlabeled data with the explicit discriminate information of classes to improve the predictive performance. However, due to the inherent additional geometry regularizer, the regularization and kernel parameters increase heavily the difficulty of model tuning. The Leave-one-out cross-validation based PRESS criterion is first presented for the solution of choosing appropriate constructing graph Laplacian parameter and geometric regularization parameters as well as semi-supervised kernel parameter. The optimization problem with two continuous and two discrete variables is solved with the proposed improved simplex search algorithm. The real-life applications validated the better classification performance of supervised learning and semi-supervised learning techniques leveraged by effectively exploiting the embedded intrinsic geometric manifolds. The local optimum problem in model optimization of MR approach are discussed.

Acknowledgements

The work was supported by National Science Foundation for Post-doctoral Scientists of China (Grant No.20090450700) and also supported by the national

key project of China: high-grade NC machine tools and fundamental manufacturing equipment (Grant No.2009ZX04014-103) and the national "863" Program of China (Grant No. 2008AA04Z801).

References

1. Zhu, X.: Semi-supervised learning literature survey, Tech. rep. (2008), http://pages.cs.wisc.edu/~jerryzhu/research/ssl/semireview.html
2. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research 7, 2399–2434 (2006)
3. Wei, L., Keogh, E.: Semi-supervised time series classification. In: KDD 2006: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 748–753. ACM, New York (2006)
4. Jansen, A., Niyogi, P.: A geometric perspective on speech sounds, Tech. rep., University of Chicago (2005)
5. Pan, J.J., Yang, Q., Chang, H., Yeung, D.Y.: A manifold regularization approach to calibration reduction for sensor-network based tracking. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence, Boston, United States, pp. 988–993 (2006)
6. Williams, O., Blake, A., Cipolla, R.: Sparse and semi-supervised visual mapping with the s^3gp. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 230–237 (2006)
7. Li, M., Xu, J., Yang, J., Yang, D., Wang, D.: Multiple manifolds analysis and its application to fault diagnosis. Mechanical Systems and Signal Processing 23(8), 2500–2509 (2009)
8. Sindhwani, V., Niyogi, P., Belkin, M.: Beyond the point cloud: from transductive to semi-supervised learning. In: ICML 2005, New York, NY, USA, pp. 824–831 (2005)
9. Chapelle, O., Zien, A.: Semi-Supervised Classification by Low Density Separation. In: AISTAT (2005)
10. Sindhwani, V.: On semi-supervised kernel methods, Ph.D. thesis, Chicago, IL, USA (2007)
11. Schölkopf, B., Smola, A.J.: Learning with Kernels. MIT Press, Cambridge (2002)
12. Cawley, G.: Leave-one-out cross-validation based model selection criteria for weighted ls-svms. In: Neural Networks, IJCNN 2006, pp. 1661–1668 (2006)
13. Allen, D.M.: The relationship between variable selection and prediction. Technometrics 16, 125–127 (1974)
14. Bo, L., Wang, L., Jiao, L.: Feature scaling for kernel fisher discriminant analysis using leave-one-out cross validation. Neural Computation 18, 961–978 (2006)
15. Yuan, J., Wang, K., Yu, T., Fang, M.: Reliable multi-objective optimization of high-speed wedm process based on gaussian process regression. International Journal of Machine Tools and Manufacture 48(1), 47–60 (2008)
16. Lagarias, J.C., Reeds, J.A., Wright, M.H., Wright, P.E.: Convergence properties of the nelder-mead simplex algorithm in low dimensions. SIAM Journal of Optimization 9, 112–147 (1996)

Probability Density Estimation Based on Nonparametric Local Kernel Regression

Min Han and Zhi-ping Liang

School of Electronic and Information Engineering, Dalian University of Technology, Dalian, Liaoning 116023, China
minhan@dlut.edu.cn

Abstract. In this research, a local kernel regression method was proposed to improve the computational efficiency after analyzing the kernel weights of the nonparametric kernel regression. Based on the correlation between the distribution function and the probability density function, together with the nonparametric local kernel regression we developed a new probability density estimation method. With the proper setting of the sparse factor, the number of the kernels involved in the kernel smooth was controlled, and the density was estimated with highly fitness and smoothness. According to the simulations, we can see that the proposed method shows a very well performance both in the accuracy and the efficiency.

Keywords: nonparametric kernel regression; probability density estimation; cumulative distribution functions.

1 Introduction

Probability density estimation from samples of unknown distributions plays an import part in the machine learning [1]. It's known to all that many pattern recognition questions, such as the classification and clustering, is based on the probability densities. And also the density estimation was an essential step of the calculation of the mutual information [2] which can be applied in many statistical areas. Therefore, it will be of great significance both in theory and practical application.

Generally speaking, both the parametric and the nonparametric methods can be used to estimate the densities, but they are quite different. The parametric method assumes that the samples are drawn from one certain distribution with the unknown parameters to be estimated. Therefore the accuracy depends much on the prior knowledge of the distribution. However, the assumption of the distribution will be much difficult in practice. On the contrary, nonparametric methods do not introduce the prior assumptions of the underlying density which characteristics are learned only from the training samples [3]. Thus, nonparametric density estimation technique has attracted much attention as it can be utilized to estimate densities functions with arbitrary shapes.

The Kernel Density Estimate (KDE) is supposed to be the most common and simple nonparametric method with high accuracy. The classical KDE can be written

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 465–472, 2010.

as a weighted average over the sample distribution function [4] with assigning equal weights for all the kernels. However, the number of kernels in the KDE is equal to the size of the training data which makes it difficult if the data size extremely large.

Much research has been done in order to improve the computational efficiency. Mark Girolami and Chao he[5] present one reduced set density estimator that provides a kernel based density estimator which employs a small percentage of the available data sample and is optimal in the L_2 sense. Recently, Chen, S. and X. Hong [6-8] proposed an even faster sparse kernel density estimate method which based on the orthogonal forward regression. But it is difficult to obtain the sparse density estimate because of the extra step of the orthogonal forward regression [9].

The aim of this research is to gain a novel density estimate method based on the nonparametric kernel regression and the cumulative distribution functions. The cumulative distribution function was calculated from the training data. Then the probability density was estimated based on the correlation with the cumulative distribution function. And lastly, the probability density estimation of highly fitness and smoothness was obtained by the improved nonparametric kernel regression. The number of the kernels involved in the kernel smooth was controlled by the settings of the sparse factor, and the efficiency was improved. The simulations of the density estimations show the ability of the method in this paper with the synthetic data.

This paper is organized as follows. In Section 2 the nonparametric kernel regression will be introduced briefly, and in Section 3 will be the proposed algorithm based on the local kernel regression and the cumulative distribution function. The numerical examples and the conclusions are given in the last two sections.

2 Nonparametric Kernel Regression

Nonparametric kernel regression has been widely used for its consonance between smoothness and fitness [10].Given a sequence (x_1, y_1), (x_2, y_2),…,(x_n, y_n) coming form a model, the aim is to estimate the nonlinear functions from the observation. The model can be expressed as the formulation:

$$y_i = m_i(x) + \varepsilon_i, \quad i = 1,...,n \tag{1}$$

where ε_i are independent, identically distributed random noise.

It is difficult to estimate the parameters through a certain model, especially when the correlation is complex. Therefore, the nonparametric regression models are used to describe the underlying relationship between the inputs and outputs.

Kernel smooth regression is a typical nonparametric regression which smoothes the value of Y by the observation data. The widely used Nadaraya-Waston estimator is described as follows:

$$\hat{m}(x) = \sum_{i=1}^{n} w_i y_i, \qquad w_i = \frac{\sum_{i=1}^{n} K_h(x - x_i)}{\sum_{i=1}^{n} K_h(x - x_i)} \tag{2}$$

where $K_h(\cdot)$ is the kernel function, and Gaussian kernels or the polynomial kernels are usually employed. In this paper, we chose the gaussian kernels showed as follows:

$$K = \exp(-\sum_{d=1}^{D} \frac{\left\| x^{(d)} \right\|^2}{2\mathbf{h}^2}) \tag{3}$$

h is the kernel bandwidth, which shows the width of the data included in the kernel smooth. Usually, one can obtain it by cross validation or the rules of thumb.

3 Density Estimation

3.1 Nonparametric Local Kernel Regression

Nadaraya–Watson estimator employs the full training data while estimating the value at one test point. In order to improve the computational efficiency, we analyzed the weights in (2). Figure 1 shows the simulation of the weights with different kernel centers.

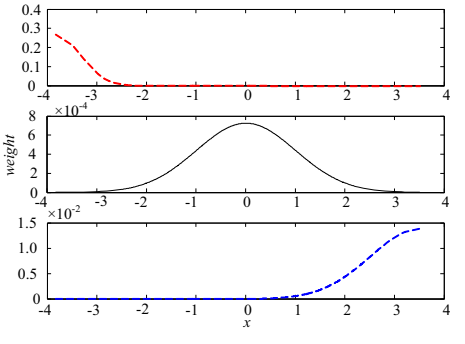


Fig. 1. Kernel weights with different centers

According to the experiment, we can see that with different kernel centers, there exist different weights. Note that, a lot of weights are zeros or near to zeros even if the kernel centers are the boundary of the data. Hence a lot of kernel computations are useless. And the computational efficiency is affected.

Therefore, in this research we propose the local kernel smooth which means that the estimate value at each point is determined by the training data next to it. The Nadaraya–Watson estimator can be improved as follows:

$$\hat{m}(\mathbf{x}) = \frac{\sum_{i-k}^{i+k} K_h(\mathbf{x} - \mathbf{x}_i) y_i}{\sum_{i-k}^{i+k} K_h(\mathbf{x} - \mathbf{x}_i)} \tag{4}$$

where parameter k is determined by the sparse factor $\alpha(0<\alpha\leq 1)$, and the value of α shows the sparseness of the kernel regression ($[\bullet]$ means to get the inter number).

$$k = [\alpha \times N] \tag{5}$$

Note that errors will be occurred while estimating weights of the boundary data points. The local kernel weights will be out of the data length. To avoid this problem we can improve the local kernel weights by formulation (6)

$$K_{i+k} = \begin{cases} K_h(\mathbf{x} - \mathbf{x}_i), & 0 < i + k \leq N \\ 0, & other \end{cases} \tag{6}$$

The computational efficiency is improved by the sparse factor which reduces the number of the kernels in the weights computation.

3.2 Density Estimation Based on Cumulative Distribution Function

The samples are usually unorganized while estimating the densities, and we can't describe the distribution clearly. Fortunately, here we can chose to estimate the empirical cumulative distribution from the data.

$$\hat{F}(x; N) = \frac{1}{N} \sum_{k=1}^{N} \prod_{j=1}^{m} \theta(x_j - x_{j,k}), \quad \theta(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases} \tag{7}$$

According to the definition of the probability density function, if the random variable X is differentiable and the derivative will be the density.

$$f(x) = F'(x) = \lim_{\Delta x \to 0^+} \frac{F(x + \Delta x) - F(x)}{\Delta x} \tag{8}$$

If Δ_x is chosen as a small number, the density can be approximated as follows:

$$f(x) \approx \frac{F(x + \Delta x) - F(x)}{\Delta x} \tag{9}$$

Therefore, the numerical result from the formulation can be chosen as an estimate of the density. However, it will not be differentiable in most cases. According to (9), what we get is the numerical results rather than the derivative, yet it can be taken as a substitute for the approximation of the density.

Note that the density estimated by the cumulative distribution function is rough. To improve the accuracy, here we can smooth the rough density with the nonparametric local kernel regression mentioned above.

The parameter h and α need to be selected beforehand while estimating the densities. Formally, the cross validation can be used to select the bandwidth. In this paper, we choose the bandwidth by a lot of experiments for simplicity.The sparse factor α determines the numbers included in the kernel weights computation. With a lot of experiments, we can get a range of [0.05, 0.35], which makes a tradeoff between the efficiency and the accuracy.

For multi-dimensional data points, the approach can be easily expended. Note that the data might be very sparse in multi-dimensional space, and hence the density estimated would be of less accuracy.

Therefore, the density estimation based on the cumulative distribution and the nonparametric local kernel regression can be summarized as follows:

1) Estimating the cumulative distribution function from the training data.
2) Then the probability density is estimated based on the correlation with the cumulative distribution function.
3) Choose the bandwidth h and the sparse factor α.
4) Smooth the rough density with the nonparametric local kernel regression.

4 Numerical Examples

Three examples were used in the simulation to test the proposed algorithm and to compare its performance with the Parzen estimator. The L_1 test error or L_2 test error was used to quantify the estimating results.

$$L_1 = \frac{1}{N_{test}} \sum_{k=1}^{N_{test}} \left| p(x_k) - \hat{p}(x_k) \right|, \quad L_2 = \frac{1}{N_{test}} \sum_{k=1}^{N_{test}} \left| p(x_k) - \hat{p}(x_k) \right|^2 \tag{10}$$

4.1 One-Dimensional Density Estimation

This was a one-dimensional example, and the density to be estimated was the mixture of eight Gaussian distributions given by

$$p(x) = \frac{1}{8} \sum_{i=0}^{7} \frac{1}{\sqrt{2\pi}\sigma_i} \mathrm{e}^{-(x-\mu_i)^2/2\sigma_i^2}, \quad \sigma_i = \sqrt{(\frac{2}{3})^i}, \ \mu_i = 3((\frac{2}{3})^i - 1), \quad 0 \le i \le 7 \tag{11}$$

The number of data points for density estimation was N=200. The experiment was repeated 200 times. The number of test data was N_{test} = 10000. Figure 2 shows a typical result with h=0.15, α=0.10. Table 1 shows the L_2 test error of each method. The proposed algorithm compares favorably with state-of-the-art kernel algorithms.

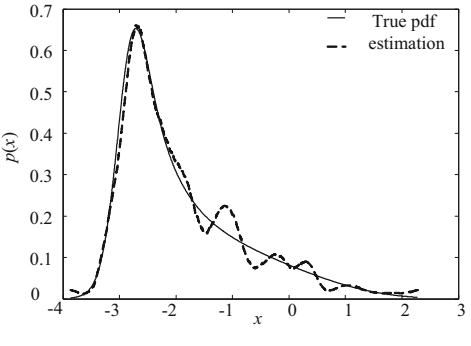


Fig. 2. One-dimensional density estimating

Table 1. The simulation result

Method	L_2 test error(mean ± standard deviation)
Prose method	$(2.5860\pm0.7382)\times10^{-3}$
Parzen KDE	$(3.0305\pm0.6903)\times10^{-3}$
SKD[8]	$(3.0181\pm2.0991)\times10^{-3}$
RSDE[5]	$(3.3\pm3.3)\times10^{-3}$

4.2 Simulation on Running Time

Another experiment was done to show the superiority of the proposed method for its running time. The density to be estimated for this one-dimensional example was the mixture of Gaussian and Laplacian given by

$$p(x)=\frac{1}{2\sqrt{2\pi}}\mathrm{e}^{-(x-2)^2/2}+\frac{0.7}{4}\mathrm{e}^{-0.7|x+2|} \tag{12}$$

Specially, the number of training data points is changed with a range from 1000 to 20000. The number of test data points was fixed at N_{test} = 1000. In the simulation, the parameters were fixed at h=0.35, α=0.15. The simulation is compared with the Prazen estimator, and the performance of the runing time was shown in Figure 3.

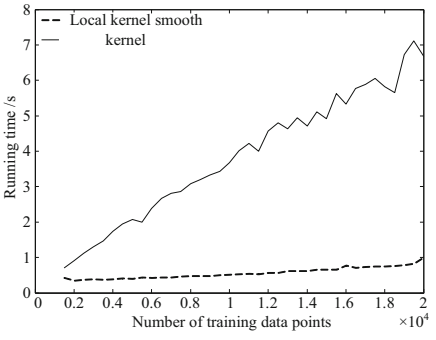


Fig. 3. Running time of the density estimation with different numbers of the training data

It can be shown that with the increment of the training data, the running time increased rapidly, while the proposed method kept at a very small running time level. And also, the L_2 test error was used to quantify the performance of the proposed method. The results were shown in Table 2. It can be concluded that computational efficiency was improved while the accuracy of the estimation unchanged.

Table 2. The comparison with the Parzen KDE

Method	L_2 test error(mean ± standard deviation)
Proposed method	$(1.1117\pm0.79760)\times10^{-5}$
Parzen KDE	$(1.1522\pm0.10429)\times10^{-5}$

4.3 Two-Dimensional Density Estimation

For this two-dimensional example, the true density to be estimated was a mixer of five Gaussion distributions.

$$p(x, y) = \sum_{i=1}^{5} \frac{1}{10\pi} \mathrm{e}^{-(x-\mu_{i,1})^2/2} \mathrm{e}^{-(y-\mu_{i,2})^2/2} \tag{13}$$

The means of the five distributions were [0.0, -4.0], [0.0, -2.0], [0.0, 0.0], [-2.0, 0.0], and [-4.0 0.0], respectively. The number of data points for estimation was 500, and the experiment was repeated 100 times. The number of the test data set was also N_{test}=10000. Figure 4 shows the true distribution and its contour plot, while Figure 5 shows the estimating results.

According to the pictures, we can see that our approach has done a very well performance. The L_1 test error was used to show the results in Table 3. For comparison the results of the typical Parzen estimate and the sparse kernel density with the same experiment conditions were quoted from [8]. It can be seen from Table 3 that for this example our approach was comparable to the both density estimates.

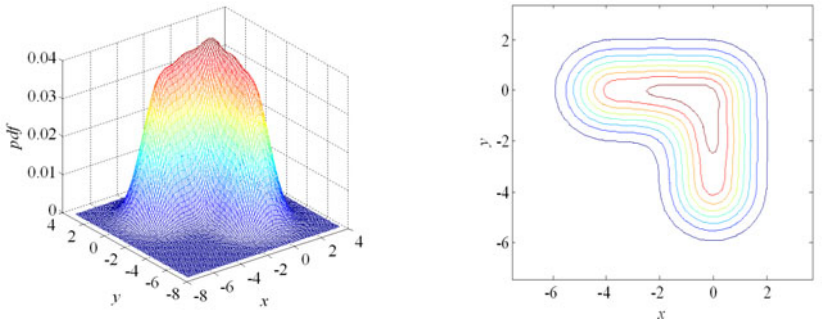


Fig. 4. True density and contour plot for the two-dimensional five Gaussian mixer

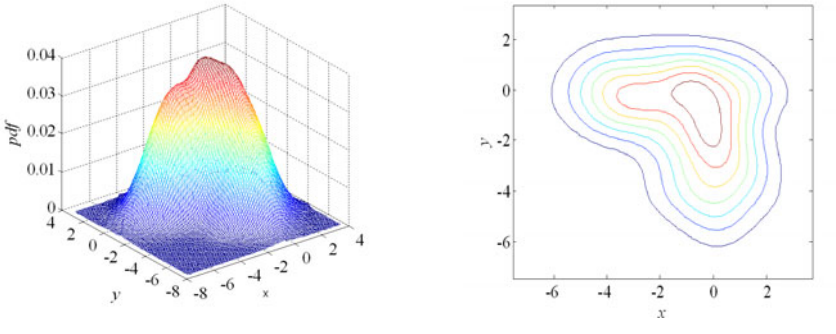


Fig. 5. The proposed density estimate and contour plot for the two-dimensional five Gaussian mixture, the bandwidth h was [0.75, 0.75], and the sparse factor: α=0.35

Table 3. The simulation result of two-dimensional density estimation

Method	L_1 test error(mean ± standard deviation)
Prose method	$(2.4978\pm0.6595)\times10^{-3}$
Parzen KDE	$(3.6204\pm0.4394)\times10^{-3}$
SKD[8]	$(3.6100\pm0.5025)\times10^{-3}$

5 Conclusion

In this research, the local kernel regression method is proposed to improve the computational efficiency of the nonparametric kernel regression. A novel probability density estimation method was introduced with the combination of the improved nonparametric kernel regression and the cumulative distribution function. The computational efficiency was improved by the selection of the sparse factor. According to the simulations, we can see that the proposed method is comparable with the classical Parzen kernel density estimation and other excellent sparse kernel density estimations.

Acknowledgments. This research is supported by the project (60674073) of the National Nature Science Foundation of China, the project (2007AA04Z158) of the National High Technology Research and Development Program of China (863 Program), the project (2006BAB14B05) of the National Key Technology R&D Program of China and the project (2006CB403405) of the National Basic Research Program of China (973 Program).

References

1. Vapnik, V.N.: The Nature of Statistical Learning Theory. Spinger, New York (1995)
2. Liu, H., Sun, J., Liu, L.: Feature selection with dynamic mutual information. Pattern Recognition 42, 1330–1339 (2009)
3. Yin, X.-f., Hao, Z.-F.: Fast kernel distribution function estimation and fast kernel density estimation on sparse bayesian learning and regularization. In: Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, Kunming, pp. 1756–1761 (2008)
4. Parzen, E.: On Estimation of a Probability Density Function and Mode. The Annals of Mathematical Statistics 33(3), 1065–1076 (1962)
5. Girolami, M., Chao, H.: Probability Density Estimation from Optimally Condensed Data Samples. IEEE Transactions on pattern analysis and machine intelligence 25(10), 1253–1264 (2003)
6. Hong, X., Chen, S., Harris, C.J.: A forward-constrained regression algorithm for sparse kernel density estimation. IEEE Transactions on Neural Networks 19(1), 193–198 (2008)
7. Chen, S., Hong, X., Harris, C.J.: Sparse kernel density construction using orthogonal forward regression with leave-one-out test score and local regularization. IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics 34(4), 1708–1717 (2004)
8. Chen, S., Hong, X., Harris, C.J.: An orthogonal forward regression technique for sparse kernel density estimation. Neurocomputing 71(4-6), 931–943 (2008)
9. Xunfu, Y., feng, H.Z.: Regularized Kernel Density Estimation Algorithm Based on Sparse Bayesian Regression. Journal of South China University of Technology (Natural Science Edition) 37(5), 23–129 (2009)
10. Zhang, J.-S., Huang, X.-F., Zhou, C.-H.: An improved kernel regression method based on Taylor expansion. Applied Mathematics and Computation 193, 419–429 (2007)

A Framework of Decision Making Based on Maximal Supported Sets

Ahmad Nazari Mohd Rose[1], Tutut Herawan[2], and Mustafa Mat Deris[3]

[1] Faculty of Informatics, Universiti Darul Iman Malaysia, Terengganu, Malaysia
[2] Department of Mathematics Education, Universitas Ahmad Dahlan, Indonesia
[3] FTMM, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia
anm@udm.edu.my, tutut81@uad.ac.id, mmustafa@uthm.edu.my

Abstract. In this paper, we present an alternative technique of decision making through parameterization reduction by determining maximal supported sets from a Boolean-valued information system based soft set theory. Based on such reduction, the maximal support will be calculated to determine the optimal choice. It is shown that the technique is identical to normal parameter reduction from previous research on soft set for decision making. While maximal support reduction is in fact has also provided consistency choices in decision making.

Keywords: Boolean-valued information system; Soft set theory; Maximal supported set; Reduction; Decision making.

1 Introduction

The reliance on information obtained from database is very critical and important. Almost in every part of our life, there are lots of instances where we have either direct or indirect dealing with databases. One aspect that database plays an important role is in the field of decision making. Input obtained from data stored in terms of records and attributes in databases do contribute a lot in the process of decision making. To this, one practical problem is faced: for a particular property, whether all the attributes in the attribute set are always necessary to preserve this property [1]. In decision making, precision is considered as a major factor. The main objective of reduction is to lessen the number of attributes, and at the same time, preserving the property of information in helping the process of decision making. The theory of soft set [2] proposed, by Molodtsov 1999 is a new method for handling uncertain data. Soft sets are called (binary, basic, elementary) neighborhood systems. As for standard soft set, it may be redefined as the classification of objects in two distinct classes, thus confirming that soft set can deal with a Boolean-valued information system. Molodtsov [2] pointed out that one of the main advantages of soft set theory is that it is free from the inadequacy of the parameterization tools, unlike in the theories of fuzzy set, probability and interval mathematics [2]. The theory of soft set has been applied to data analysis and decision support systems. The concept of reducts is

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 473–482, 2010.

another area which purportedly supports decision making with less involvement of data and attributes by reducing the attributes. By applying the concept of co-occurrence of parameters in an object and it's support on the Boolean-valued information based on soft set theory, we propose an alternative technique termed as maximal supported sets reduct. The main purpose of the propose technique is to ensure that any process of attribute elimination in transforming the complex database into much simpler database for decision making but not at the expense that will cause changes to the optimal and sub-optimal choices.

The rest of this paper is organized as follows. Section 2 describes related works of reduction and decision making under soft set theory. Section 3 describes the fundamental concept of an information system and soft set theory. Section 4 analyses the work that has been done by Maji [3], Chen *et al.* [4] and Kong *et al.* [5]. Section 5 will introduce an alternative technique of reduction and decision making based on maximal supported sets. Section 6 describes experimental result. Finally, we conclude our works in section 7.

2 Related Works

The idea of reduct and decision making using soft set theory is firstly proposed by Maji *et al.* [3]. In [3], the application of soft set theory to a decision making problem with the help of Pawlak's rough mathematics was presented. Decision was selected from among the objects based on maximal weighted value, and it is only obtained by reducing the attributes using Pawlak's rough reduction. However Chen *et al.* [4] highlighted the incorrectness out of the results used by the reduction proposed by Maji *et al.* [3]. Chen *et al.* [4] also did not miss to point out what was inappropriate in Maji's algorithm used to derive the choice value in selecting the optimal objects for the decision problem. They also pointed out that the idea of reduct under rough set theory generally cannot be applied directly in reduct under soft set theory. The idea of parameterization reduction of soft sets by Chen *et al.* [4] for soft set reduction is only focused on the optimal choice related to each object. However, the idea proposed by Chen *et al.* [4] cannot be considered complete, since the problems of the sub-optimal choice have not been addressed. To this, Kong *et al.* [5] analyzed the problem of suboptimal choice and has added parameter set of soft set. Then, they introduced the definition of normal parameter reduction in soft set theory to overcome the problems in Chen *et al.* [4] 's model by describing two new definitions, i.e. parameter important degree and soft decision partition and has used them to analyze the algorithm of normal parameter reduction. With this technique, the optimal and sub-optimal choices are still been upheld. Our paper will present that our proposed alternative technique of reduct based on maximal supported sets is better than the proposed attribute reduction by Maji *et al.* [3] and parameter reduction by Chen *et al.* [4] and at par in terms of achievement with normal parameter reduction proposed by Kong *et al.* [5].

3 Preliminaries

3.1 Information System

An *information system* is a 4-tuple $S=(U,A,V,f)$, where $U=\{u_1,u_2,\cdots,u_{|u|}\}$ is a non-empty finite set of objects, $A=\{a_1,a_2,\cdots,a_{|A|}\}$ is a non-empty finite set of attributes, $V=\bigcup_{e_i\in A}V_{e_i}$, V_{e_i} is the domain (value set) of attribute a, $f:U\times A\to V$ is an information function such that $f:U\times A\to V$, $f(u,a)\in V_a$, for every $(u,a)\in U\times A$, called information (knowledge) function. In an information system $S=(U,A,V,f)$, if $V_a=\{0,1\}$ for every $a\in A$, then S is called a *Boolean-valued information system*.

3.2 Soft Set Theory

Throughout this section U refers to an initial universe, E is a set of parameters, $P(U)$ is the power set of U.

Definition 1. (See [2].) *A pair* (F,E) *is called a soft set over U, where F is a mapping given by* $F:E\to P(U)$

In other words, a soft set is a parameterized family of subsets of the universe U. For $\varepsilon\in E$, $F(\varepsilon)$ may be considered as the set of ε-elements of the soft set (F,E)or as the set of ε-approximate elements of the soft set, instead of a (crisp) set.

Example 2. As an illustration, let we consider a soft set (F,E) which describes the "attractiveness of credit card promotions" that Mr. X is considering to purchase. Let assume that there are thirty credit card promotions in the universe U that are under consideration, $U=\{p_1,p_2,\cdots,p_{30}\}$, and E is a set of decision parameters, $E=\{e_1,e_2,e_3,e_4,e_5,e_6,e_7\}$, where e_1 stands for the parameter "Magazine", e_2 stands for the parameter "Watch", e_3 stands for the parameter "Life", e_4 stands for the parameter "Credit Card Insurance", e_5 stands for the parameter "Car", e_6 stands for the parameter "Car Insurance" and e_7 stands for the parameter "House". Consider the mapping $F:E\to P(U)$ given by "credit card promotions $(\cdot)$", where $(\cdot)$ is to be filled in by one of parameters $e\in E$. Suppose that $F(e_1)=\{p_1,p_2,p_3,p_4,p_5,p_6,p_7,p_8,p_9,p_{10},p_{11},p_{12},p_{13},p_{14},p_{15},p_{16},p_{17},p_{18},p_{19},p_{20},p_{21},p_{22},p_{25},p_{27},p_{29}\}$, $F(e_2)=\{p_2,p_3,p_{11},p_{15},p_{16},p_{18},p_{23},p_{24},p_{26},p_{28},p_{30}\}$, $F(e_3)=\{p_2,p_3,p_4,p_7,p_{11},p_{12},p_{15},p_{16},p_{18},p_{19},p_{23},p_{24},p_{26},p_{28},p_{30}\}$, $F(e_4)=\{p_2,p_3,p_{15},p_{16},p_{18}\}$, $F(e_5)=\{p_1,p_4,p_7,p_{12},p_{13},p_{14},p_{19}\}$, $F(e_6)=\{p_4,p_7,p_{11},p_{12},p_{19},p_{23},p_{24},p_{26},p_{28},p_{30}\}$, $F(e_7)=\{p_1,p_2,p_3,p_5,p_6,p_8,p_9,p_{10},p_{13},p_{14},p_{15},p_{16},p_{17},p_{18},p_{20},p_{21},p_{22},p_{25},p_{27},p_{29}\}$. As for example, $F(e_2)$ means credit card promotion for watch, whose functional value is the set $\{p_2,p_3,p_{11},p_{15},p_{16},p_{18},p_{23},p_{24},p_{26},p_{28},p_{30}\}$.

Thus, we can view the soft set (F,E) as a collection of approximations as illustrated below:

$$(F,E)=\left\{\begin{array}{l} e_1=\{p_1,p_2,p_3,p_4,p_5,p_6,p_7,p_8,p_9,p_{10},p_{11},p_{12},p_{13},p_{14},p_{15},\\ \quad p_{16},p_{17},p_{18},p_{19},p_{20},p_{21},p_{22},p_{25},p_{27},p_{29}\},\\ e_2=\{p_2,p_3,p_{11},p_{15},p_{16},p_{18},p_{23},p_{24},p_{26},p_{28},p_{30}\},\\ e_3=\{p_2,p_3,p_4,p_7,p_{11},p_{12},p_{15},p_{16},p_{18},p_{19},p_{23},p_{24}p_{26},p_{28},p_{30}\},e_4=\{p_2,p_3,p_{12},p_{16},p_{18}\},\\ e_5=\{p_1,p_4,p_7,p_{12},p_{13},p_{14},p_{19}\},e_6=p_4,p_7,p_{11},p_{12},p_{19},p_{23},p_{24},p_{26},p_{28},p_{30}\},\\ e_7=\{p_1,p_2,p_3,p_5,p_6,p_8,p_9,p_{10},p_{13},p_{14},p_{15},p_{16},p_{17},p_{18},p_{20},p_{21},p_{22},p_{25},p_{27},p_{29}\} \end{array}\right\}$$

Fig. 1. The soft set

The above soft set can be represented as a finite table, where the entries are "1" which is denoted as the presence of described attribute and "0" which mean the attribute is not part of the description of the house. The table can also be viewed as a Boolean-valued information system. The relation between a soft set and a Boolean-valued information system is given as follow.

Proposition 3. *If* (F,E) *is a soft set over the universe U, then* (F,E) *is a binary-valued information system* $S=(U,A,V_{\{0,1\}},f)$.

Proof. Let (F,E) be a soft set over the universe *U*, we define a mapping

$$F=\{f_1,f_2,\cdots,f_n\},$$

where

$$f_i:U\to V_i \text{ and } f_i(x)=\begin{cases}1, & x\in F(e_i)\\ 0, & x\notin F(e_i)\end{cases}, \text{ for } 1\le i\le |A|.$$

Hence, if $A=E$, $V=\bigcup_{e_i\in A}V_{e_i}$, where $V_{e_i}=\{0,1\}$, then a soft set (F,E) can be considered as a binary-valued information system $S=(U,A,V_{\{0,1\}},f)$. □

From Proposition 3, it is easily to understand that a binary-valued information system can be represented as a soft set. Thus, we can make a one-to-one correspondence between (F,E) over *U* and $S=(U,A,V_{\{0,1\}},f)$.

4 Parameterization Reduction and Decision Making

In this section, we present the existing techniques on soft reduction and decision making techniques proposed by [3], [4] and [5]. The purpose of this analysis is to present the comparison in the previous techniques and how our proposed technique will provide an alternative way for soft decision making. Suppose we have a soft set (F,E) over universe *U* with the representation as displayed in Figure 1. Let $f_E(p_i)=\sum_j p_{ij}$, where p_{ij} are the entries in the Boolean-table of (F,E). Since it is

clearly shown that $f_E(p_i)_{i=2,3,15,16,18} = 5$ is the maximum choice value, thus p_2, p_3, p_{15}, p_{16}, and p_{18} are the original (not reduced) optimal choice.

4.1 Parameterization Reduction of Maji *et al.* [3]

Maji *et al.* [3] presented a reduction of soft sets and its applications in a decision making problem, which can be briefly explained as follows. The most optimal decision derived by Maji *et al.* [3] will be only be deduced after the process of identifying the rough set-based reduction set. This can by maintained using a partition on *U* based on the indiscernibility relation on a set of attributes in rough set theory [6]. As for our example based on Figure 1, the partition induced by the set of all attributes $E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$, denoted by U/E is given by

$$U/E = \left\{ \begin{matrix} \{p_1, p_{13}, p_{14}\}, \{p_2, p_3, p_{15}, p_{16}, p_{18}\}, \{p_4, p_7, p_{12}, p_{19}\}, \{p_5, p_6, p_8, p_9, p_{10}, \\ p_{17}, p_{20}, p_{21}, p_{22}, p_{25}, p_{27}, p_{29}\}, \{p_{11}, p_{23}, p_{24}, p_{26}, p_{28}, p_{30}\} \end{matrix} \right\},$$

and optimal choices are now $\{p_2, p_3, p_{15}, p_{16}, p_{18}\}$, denoted by the maximum value = 5. Any subset of *E*, that will induce partition equal to U/E will be considered as attribute reduction of E. As for example, let $R \subset E$, where $R = \{e_1, e_2, e_5, e_7\}$, it will produce a partition induced as follow

$$U/R = \left\{ \begin{matrix} \{p_1, p_{13}, p_{14}\}, \{p_2, p_3, p_{15}, p_{16}, p_{18}\}, \{p_4, p_7, p_{12}, p_{19}\}, \{p_5, p_6, p_8, p_9, p_{10}, \\ p_{17}, p_{20}, p_{21}, p_{22}, p_{25}, p_{27}, p_{29}\}, \{p_{11}, p_{23}, p_{24}, p_{26}, p_{28}, p_{30}\} \end{matrix} \right\}.$$

Therefore, *R* can also be considered as a reduct of *E*. However, the optimal objects are now $\{p_1, p_2, p_3, p_{13}, p_{14}, p_{15}, p_{16}, p_{18}\}$, which has the maximum value of 3. The optimal choices obtained from this set *R* are different from the set *E*, which is $\{p_2, p_3, p_{15}, p_{16}, p_{18}\}$. In the other hand, after reduction, the choices for sub-optimal are the set $\{p_4, p_7, p_5, p_6, p_8, p_9, p_{10}, p_{12}, p_{17}, p_{19}, p_{20}, p_{21}, p_{22}, p_{25}, p_{27}, p_{29}\}$, which is different from sub-optimal choices derived from *E*, which is $\{p_4, p_7, p_{12}, p_{19}\}$. The major drawback is inconsistency as shown in the selection of optimal and sub-optimal chaices.

4.2 Soft Parameter Reduction of Chen *et al.* [4]

Chen *et al.* [4] has defined $f_E(p_i) = \sum_j p_{ij}$, where p_{ij} are the entries in the table of (F, E) and and M_E denoted for collection of objects in *U* which has the maximum value of f_E. Chen *et al.* [4] has in fact defined *a dispensable set* $A \subset E$ if only if $M_{E \setminus A} = M_E$. Clearly, parameter reduction of Chen *et al.* [4] has been able to provide consistency in optimal object's decision. In our analysis based on Figure 1, $M_E = \{p_2, p_3, p_{15}, p_{16}, p_{18}\}$. Let $S = \{e_3, e_4, e_5, e_6, e_7\}$, thus we have

$$M_{E-\{e_3,e_4,e_5,e_6,e_7\}} = M_{\{e_1,e_2\}} = \{p_2, p_3, p_{15}, p_{16}, p_{18}\}.$$

Thus, the set $S = \{e_3, e_4, e_5, e_6, e_7\}$ is dispensable.

Optimal choices are still $\{p_2, p_3, p_{15}, p_{16}, p_{18}\}$ which has the maximum value of 2. and $M_{E-\{e_3,e_4,e_5,e_6,e_7\}} = M_{\{e_1,e_2\}}$. Therefore, $\{e_1, e_2\}$ can be considered as a parameter reduction of *E*. Chen *et al.* [4] has succesfully maintain consistency in the optimal choices, but failed in maintaining consistency for sub-optimal choices. Notice that, in this case, the sub-optimal choices are all of the promotions except the optimal promotions, i.e., $\{p_1, p_2, p_3, p_5, p_6, p_8, p_9, p_{10}, p_{11}, p_{13}, p_{14}, p_{15}, p_{16}, p_{17}, p_{18}, p_{20}, p_{21}, p_{22}, p_{23}, p_{24}, p_{25}, p_{26}, p_{27}, p_{28}, p_{29}, p_{30}\}$, which is different from sub-optimal choices as derived from *E*, i.e., $\{p_4, p_7, p_{12}, p_{19}\}$. The major drawback here, is again inconsistency as made known in the selection of sub-optimal choices.

4.3 Normal Parameter Reduction of Kong *et al.* [5]

The main purpose of Kong *et al.*'s normal parameter reduction is to provide consistency in selecting the optimal and sub-optimal objects from any reduced set that does conforms to the rules that has been defined by Kong *et al.* [5]. Kong has maintained the same the partitions of objects by defining *indiscernibility relation* $IND(A)$, for $A \subset E$ as follows:

$$IND(A) = \{(p_i, p_j) \in U \times U : f_A(p_i) = f_A(p_j)\}.$$

The *decision partition* of *U* generated by $IND(E)$ is defined as

$$C_E = \left\{\{p_1, \cdots, p_i\}_{f_1}, \{p_{i+1}, \cdots, p_j\}_{f_2}, \cdots, \{p_k, \cdots, p_n\}_{f_s}\right\}.$$

In the case that for $A \subset E$, if $f_A(p_1) = f_A(p_2) = ... = f_A(p_n)$ implies $C_E = C_{E-A}$, then *A* is called *dispensable set*. For this definition, Kong *et al.* [5] has termed $E - A$ as normal parameter reduction. Based on on the Boolean table in Figure 1, the decision partition induced, will be

$$\left\{\begin{matrix}\{p_2, p_3, p_{15}, p_{16}, p_{18}\}_5, \{p_4, p_7, p_{12}, p_{19}\}_4, \{p_1, p_{11}, p_{13}, p_{14}, p_{23}, p_{24}, p_{26}, p_{28}, p_{30}\}_3 \\ \{p_5, p_6, p_8, p_9, p_{10}, p_{17}, p_{20}, p_{21}, p_{22}, p_{25}, p_{27}, p_{29}\}_2\end{matrix}\right\}$$

and $\{p_2, p_3, p_{15}, p_{16}, p_{18}\}$ is the optimal objects and $\{p_4, p_7, p_{12}, p_{19}\}$ will be the sub-optimal objects. Let a subset, $Z = \{e_6, e_7\} \subset E$. Obviously, *Z* is dispensable since the decision partition generated by *Z* has not changed, that is $C_E = C_{E-Z}$. Thus, by deleting parameter *Z* from *E*, we will have as what so-called normal parameterization reduction. And Kong *et al.* has successfully shown that the optimal and sub-optimal decisions, thus maintaining consistency in decision after the reduction.

5 Soft Decision Making Using Maximal Supported Sets

In this section, we propose an alternative technique of obtaining the optimal and sup-optimal choices in soft reduction and decision making. It is based on maximal supported sets from a Boolean table. Firstly, by using the notion of co-occurrence of parameters on an object, we then define the notion of support of an object. Then, the reduct will be obtained based on the maximum support set value. The next nearest support set value will be the indicator for the sub-optimal value.

Throughout this sub-section the pair (F,E) refers to the soft set over the universe U representing a Boolean-valued information system $S=(U,A,V_{\{0,1\}},f)$.

Definition 4. *Let (F,E) be a soft set over the universe U and $u\in U$. A parameter co-occurrence set of an object u can be defined as* $\mathrm{coo}(u)=\{e\in E: f(u,e)=1\}$.

Obviously, $\mathrm{Coo}(u)=\{e\in E: F(e)=1\}$.

Definition 5. *Let (F,E) be a soft set over the universe U and $u\in U$. Support of an object u is defined by* $\mathrm{supp}(u)=\mathrm{card}(\{e\in E: f(u,e)=1\})$.

Definition 6. *Let (F,E) be a soft set over the universe U and $u\in U$. An objects u is said to be maximally supported by a set of all parameters E, denoted by* $\mathrm{Msupp}(u)$ *if* $\mathrm{supp}(u)>\mathrm{supp}(v),\ \forall v\in U\setminus\{u\}$.

Based on Definition 6, we can make supported (ranked) ordered objects according their support value as $U_1>U_2>\cdots>U_n$, where $U_i\subseteq U$ and $U_i=\{u\in U: u \text{ is } i\text{ - th maximal supported by } E\}$, for $1\le i\le n$. Thus, U_i is a collection of objects in U having the same support, i.e., objects of the same support of are grouped into the same class. Obviously $U=\bigcup_{1\le i\le n}U_i$ and $U_i\cap U_j=\phi$, for $i\neq j$. In other word, a collection of $U/E=\{U_1,U_2,\cdots,U_n\}$ is a decision partition of U, so called *cluster decision* of U.

Definition 7. *Let (F,E) be a soft set over the universe U and $A\subset E$. A is said to be indispensable if $U/A=U/E$. Otherwise, A is said to be dispensable.*

Based on Definition 7, we can reduce the number of parameters without changing the optimal and sub-optimal decisions.

Definition 8. *For soft set (F,E) over the universe U and $A\subseteq E$. A is reduction of E if only if A is indispensable and* $\mathrm{supp}_{E\setminus A}(u)=_{E\setminus A}\mathrm{supp}(v)$, *for every* $u,v\in U$.

Definition 9. *For soft set (F,E) over the universe U and $u\in U$. An object u will be the optimal decision if u is maximally supported by E.*

The pseudo-code of searching reduct based on using soft set theory based on maximal supported sets is as follows.

1. Input a soft set (F,E) over a universe U, where E as the available parameters used for the description of U and a representation of (F,E) in a Boolean-valued information system $S=(U,A,V_{\{0,1\}},f)$.
2. Determine the co-occurrence of parameters on each object and calculate its support.
3. Determine the first, second until the least support.
4. From step 3, determine the cluster decision partition U/E.
5. Determine $A\subset E$, such that $U/A=U/E$ and $\text{supp}_A(u)=_A \text{supp}(v)$, for every $u,v\in U$.
6. The subset A will be identified as the reduct preserving the optimal and sub-optimal decisions.

Fig. 2. The pseudo-code of the proposed technique

6 Result and Discussion

The following will be the co-occurrence set derived from Boolean-valued information of Figure 1.

$$\text{Coo}(p_1)=\{e_1,e_5,e_7\},\ \text{Coo}(p_2)=\{e_1,e_2,e_3,e_4,e_7\},\ \text{Coo}(p_3)=\{e_1,e_2,e_3,e_4,e_7\},$$
$$\text{Coo}(p_4)=\{e_1,e_3,e_5,e_6\},\ \text{Coo}(p_5)=\{e_1,e_7\},\text{Coo}(p_6)=\{e_1,e_7\},$$
$$\text{Coo}(h_7)=\{e_1,e_3,e_5,e_6\},\ \text{Coo}(p_8)=\{e_1,e_7\},\ \text{Coo}(p_9)=\{e_1,e_7\},$$
$$\text{Coo}(p_{10})=\{e_1,e_7\},\ \text{Coo}(p_{11})=\{e_2,e_3,e_6\},\ \text{Coo}(p_{12})=\{e_1,e_3,e_5,e_6\},$$
$$\text{Coo}(p_{13})=\{e_1,e_5,e_7\},\ \text{Coo}(p_{14})=\{e_1,e_5,e_7\},\ \text{Coo}(p_{15})=\{e_1,e_2,e_3,e_4,e_7\},$$
$$\text{Coo}(p_{16})=\{e_1,e_2,e_3,e_4,e_7\},\ \text{Coo}(p_{17})=\{e_1,e_7\},\ \text{Coo}(p_{18})=\{e_1,e_2,e_3,e_4,e_7\},$$
$$\text{Coo}(p_{19})=\{e_1,e_3,e_5,e_6\},\ \text{Coo}(p_{20})=\{e_1,e_7\},\ \text{Coo}(p_{21})=\{e_1,e_7\},$$
$$\text{Coo}(p_{22})=\{e_1,e_7\},\ \text{Coo}(p_{23})=\{e_2,e_3,e_6\},\ \text{Coo}(p_{24})=\{e_2,e_3,e_6\},$$
$$\text{Coo}(p_{25})=\{e_1,e_7\},\ \text{Coo}(p_{26})=\{e_2,e_3,e_6\},\ \text{Coo}(p_{27})=\{e_1,e_7\},$$
$$\text{Coo}(p_{28})=\{e_2,e_3,e_6\},\ \text{Coo}(p_{29})=\{e_1,e_7\},\ \text{Coo}(p_{30})=\{e_2,e_3,e_6\},$$

Thus, support each transaction is given as follow

$$\text{supp}(p_i)=5,\quad i=2,3,15,16,18$$
$$\text{supp}(p_j)=4,\quad j=4,7,12,19$$
$$\text{supp}(p_k)=3,\quad k=1,11,13,14,23,24,26,28,30$$
$$\text{supp}(p_l)=2,\quad l=5,6,8,9,10,17,20,21,22,25,27,29$$

Therefore, the cluster partition is

$$U/E=\left\{\begin{matrix}\{p_2,p_3,p_{15},p_{16},p_{18}\},\{p_4,p_7,p_{12},p_{19}\},\{p_1,p_{11},p_{13},p_{14},p_{23},p_{24},p_{26},p_{28},p_{30}\}\\ \{p_5,p_6,p_8,p_9,p_{10},p_{17},p_{20},p_{21},p_{22},p_{25},p_{27},p_{29}\}\end{matrix}\right\},$$

where it is arrange in descending order of support value. As noted from such cluster, the maximum supported set is $\{p_2, p_3, p_{15}, p_{16}, p_{18}\}$ with the support of each element is 5. And $\{p_4, p_7, p_{12}, p_{19}\}$ can be considered as the sub-optimal set based on the support value which the next highest, i.e. 4. By comparing optimal choices from our proposed technique with parameter reduction from [5], also giving the result of the same the optimal and sub-optimal choices. Thus confirming that our reduction also provide the right choice for decision making. As in [3] and [4], only the issue of optimal choice was addressed, but in our paper, any set of reduct that conforms to our rule of reduct will still provide the same optimal and sub-optimal. To elaborate Definitions 7 and 8, let $A = \{e_1, e_2, e_3, e_4, e_5\}$. Then we will obtain $\text{supp}_{E\setminus A}(p_i) = 1, \quad 1 \le i \le 30$. Therefore A is indispensable, and we can now delete parameters e_6 and e_7 from E. By deleting e_6 and e_7, we now have

$$U/A = \left\{ \begin{matrix} \{p_2, p_3, p_{15}, p_{16}, p_{18}\}, \{p_4, p_7, p_{12}, p_{19}\}, \{p_1, p_{11}, p_{13}, p_{14}, p_{23}, p_{24}, p_{26}, p_{28}, p_{30}\} \\ \{p_5, p_6, p_8, p_9, p_{10}, p_{17}, p_{20}, p_{21}, p_{22}, p_{25}, p_{27}, p_{29}\} \end{matrix} \right\},$$

which is still the same partition as in U/E. As we can see from $U/E = U/A$, the partition is invariant, therefore $\{e_6, e_7\}$ can be deleted from E. Also in the case of $U/E = U/A$, the maximum supported sets are still maintained. Again, the optimal and sub-optimal choices has not changed, thus confirming that eliminating technique based on parameter of co-occurrence and supported sets will not change the optimal and sub-optimal choices. So this where, our proposed reduction technique is better then the one proposed by [3] and [4], because it still maintains the same optimal and sub-optimal choices after reduction. Thus, it is shown that our proposed reduction technique is at par with normal parameter reduction by [5] at the point of selecting optimal and sub-optimal choices.

7 Conclusion

In this paper, we have presented an alternative technique of decision making based on algorithm termed maximal supported sets. In order to apply this technique, a Boolean-valued information system is required. Since the "standard" soft set deals with such information system, thus maximal supported sets reduction can be applied for the process of decision making. Using the co-occurrence of parameters concept in an object, we define the notion of a support of an object under soft set theory. Based on the supported set from the parameters co-occurrence, we are able to identify optimal and sub-optimal choices. It is proven to be better than reduction of [3] and [4] because we are still able to maintain the optimal and sub-optimal objects after the reduction. Our reduction technique has proven that the optimal choice has never changed after reduction and different reductions will still decide the same optimal and sub-optimal choices. Thus proving our proposed reduction provided much improved and consistency in decision making. Therefore, it is safe to say that our proposed technique will not give rise misleading final decision.

References

1. Zhao, Y., Luo, F., Wong, S.K.M., Yao, Y.Y.: A general definition of an attribute reduct. In: Yao, J., Lingras, P., Wu, W.-Z., Szczuka, M.S., Cercone, N.J., Ślęzak, D. (eds.) RSKT 2007. LNCS (LNAI), vol. 4481, pp. 101–108. Springer, Heidelberg (2007)
2. Molodtsov, D.: Soft set theory-first results. Computers and Mathematics with Applications 37, 19–31 (1999)
3. Maji, P.K., Roy, A.R., Biswas, R.: An application of soft sets in a decision making problem. Compututer and Mathematics with Application 44, 1077–1083 (2002)
4. Chen, D., Tsang, E.C.C., Yeung, D.S., Wang, X.: The Parameterization Reduction of Soft Sets and its Applications. Computers and Mathematics with Applications 49, 757–763 (2005)
5. Kong, Z., Gao, L., Wang, L., Li, S.: The normal parameter reduction of soft sets and its algorithm. Computers and Mathematics with Applications 56, 3029–3037 (2008)
6. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Sciences 177(1), 3–27 (2007)

Dynamics of Competitive Neural Networks with Inverse Lipschitz Neuron Activations

Xiaobing Nie and Jinde Cao

Department of Mathematics, Southeast University, Nanjing 210096, China
{xbnie,jdcao}@seu.edu.cn

Abstract. In this paper, by using nonsmooth analysis approach, topological degree theory and Lyapunov-Krasovskii function method, the issue of global exponential stability is investigated for competitive neural networks possessing inverse Lipschitz neuron activations. Several novel sufficient conditions are established towards the existence, uniqueness and global exponential stability of the equilibrium point for competitive neural networks with time-varying delay.

Keywords: Competitive neural networks, Inverse Lipschitz neuron activations, Global exponential stability.

1 Introduction

In this paper, we consider the following competitive neural networks (**CNNs**) with time-varying delay:

$$\begin{cases} \dfrac{\mathrm{d}x_i(t)}{\mathrm{d}t} = -\alpha_i(x_i(t)) + \sum\limits_{j=1}^{N} D_{ij} f_j(x_j(t)) + \sum\limits_{j=1}^{N} D_{ij}^{\tau} f_j(x_j(t-\tau(t))) \\ \qquad\qquad +B_i S_i(t) + I_i, \\ \dfrac{\mathrm{d}S_i(t)}{\mathrm{d}t} = -S_i(t) + c f_i(x_i(t)), \end{cases} \tag{1}$$

where $i = 1, 2, \cdots, N$, c is a constant and the time delay $\tau(t)$ is a time-varying differentiable function satisfies

$$0 \le \tau(t) \le \tau, \quad \dot{\tau}(t) \le \mu. \tag{2}$$

System (1) can be rewritten as the following vector form:

$$\begin{cases} \dfrac{\mathrm{d}x(t)}{\mathrm{d}t} = -\alpha(x(t)) + Df(x(t)) + D^{\tau} f(x(t-\tau(t))) + BS(t) + I, \\ \dfrac{\mathrm{d}S(t)}{\mathrm{d}t} = -S(t) + cf(x(t)), \end{cases} \tag{3}$$

where $x = (x_1, x_2, \cdots, x_N)^T \in \mathbb{R}^N$, $S = (S_1, S_2, \cdots, S_N)^T \in \mathbb{R}^N$, $\alpha(x(t)) = \mathrm{diag}(\alpha_1(x_1(t)), \alpha_2(x_2(t)), \cdots, \alpha_N(x_N(t)))$, $D = (D_{ij})_{N\times N}$, $D^{\tau} = (D_{ij}^{\tau})_{N\times N}$, $B = \mathrm{diag}(B_1, B_2, \cdots, B_N)$, $f(x(t)) = (f_1(x_1(t)), f_2(x_2(t)), \cdots, f_N(x_N(t)))^T$,

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 483–492, 2010.

$f(x(t-\tau(t))) = (f_1(x_1(t-\tau(t))), f_2(x_2(t-\tau(t))), \cdots, f_N(x_N(t-\tau(t))))^T$, $I = (I_1, I_2, \cdots, I_N)^T$.

Recently, the stability analysis of **CNNs** with their various generalizations has attracted the attention of many researchers, see for example [1-6]. However, it is worth noting that in most existing stability results for competitive neural networks, there is a basic assumption: the neuron activation functions are Lipschitz continuous, and/or monotonic increasing [1-5]. In this paper, it is our purpose to study system (1) with (2) and inverse Lipschitz neuron activations. Firstly, we present a sufficient condition for checking the existence, uniqueness of the equilibrium point for system (1) by applying topological degree theory. Secondly, by employing the continuous theorem and Lyapunov-Krasovskii function method, a delay-dependent criterion guaranteeing the **GES** of the unique equilibrium point is derived. The obtained criteria are shown in terms of LMIs and based on nonsmooth analysis approach, linear matrix inequality technique.

Notations. For any matrix A, A^T stands for the transpose of A. If A is a symmetric matrix, $A < 0\,(A \leq 0)$ means that A is negative definite (negative semidefinite). $\lambda_{\min}(A)$ represents the minimum eigenvalue of matrix A. Given the column vector, $x = (x_1, x_2, \cdots, x_n)^T \in \mathbb{R}^n$, the norm is the Euclidean vector norm, i.e., $\|x\| = (\sum_{i=1}^{n} x_i^2)^{1/2}$. $\dot{x}(t)$ denotes the derivative of function $x(t)$.

2 Preliminaries

The initial conditions associated with system (1) are of the form

$$\begin{aligned} x_i(t) &= \phi_i(t), \qquad t \in (-\tau, 0], \\ S_i(t) &= \varphi_i(t) \equiv \varphi_i(0), \; t \in (-\tau, 0]. \end{aligned}$$

Definition 1 [7, 8, 9]. A continuous function $g : \mathbb{R} \to \mathbb{R}$ is said to be $\alpha-$inverse Lipschitz, if

(i) g is a strictly monotonic increasing function;
(ii) For any $\rho \in \mathbb{R}$, there exist constants $q_\rho > 0$ and $r_\rho > 0$ which depend on ρ, satisfying

$$|g(\theta) - g(\rho)| \geq q_\rho |\theta - \rho|^\alpha, \quad \forall\, |\theta - \rho| \leq r_\rho,$$

where $\alpha > 0$ is a constant.

By $\mathcal{IL}(\alpha)$, we denote the class of $\alpha-$inverse Lipschitz functions. There are a great number of functions which belong to $\mathcal{IL}(\alpha)$. For example, $g(\theta) = \theta^3 \in \mathcal{IL}(3)$, $g(\theta) = \arctan\theta \in \mathcal{IL}(1)$.

Lemma 1([8]). If $g(\theta) \in \mathcal{IL}(\alpha)$, then for any $\rho_0 \in \mathbb{R}$, we have

$$\lim_{|\rho| \to +\infty} \int_{\rho_0}^{\rho} [g(\theta) - g(\rho_0)] \mathrm{d}\theta = +\infty. \tag{4}$$

Lemma 2([8]). If $g(\theta) \in \mathcal{IL}(\alpha)$ and $g(0) = 0$, then there exist constants $q_0 > 0$ and $r_0 > 0$, such that

$$|g(\theta)| \geq q_0 |\theta|^\alpha, \quad \forall\, |\theta| \leq r_0. \tag{5}$$

Moreover,

$$|g(\theta)| \geq q_0\, r_0^{\alpha}, \quad \forall\, |\theta| \geq r_0. \tag{6}$$

Lemma 3([10]). Let function $F : \mathbb{R}^n \to \mathbb{R}^n$ be locally Lipschitz continuous. For any given $x, y \in \mathbb{R}^n$, there exists an element W in the union $\bigcup_{z\in[x,y]} \partial F(z)$ such that

$$F(y) - F(x) = W(y - x),$$

where $[x, y]$ denotes the segment connecting x and y, ∂F denotes the generalized Jacobian of function F.

Lemma 4. For any vectors $x,\ y \in \mathbb{R}^n$ and positive definite matrix $G \in \mathbb{R}_{n\times n}$, the following matrix inequality holds:

$$2\, x^T y \leq x^T G x + y^T G^{-1} y.$$

Let Ω be a nonempty, bounded and open subsets of $\mathbb{R}^n$. The closure and boundary of Ω are denoted by $\overline{\Omega}$ and $\partial\Omega$, respectively.

Lemma 5([11]). Let $H(\lambda, x) : [0, 1] \times \overline{\Omega} \to \mathbb{R}^n$ be a continuous homotopic mapping. If $H(\lambda, x) = y$ has no solutions in $\partial\Omega$ for $\lambda \in [0, 1]$ and $y \in \mathbb{R}^n \setminus H(\lambda, \partial\Omega)$, then the topological degree $\deg(H(\lambda, x), \Omega, y)$ of $H(\lambda, x)$ is a constant which is independent of λ. In this case, $\deg(H(0, x), \Omega, y) = \deg(H(1, x), \Omega, y)$.

Lemma 6([11]). Let $H(x) : \overline{\Omega} \to \mathbb{R}^n$ be a continuous mapping. If $\deg(H(x), \Omega, y) \neq 0$, then there exists at least one solution of $H(x) = y$ in Ω.

Throughout this paper, we make the following assumptions.

(A_1) Functions $\alpha_i : \mathbb{R} \to \mathbb{R}$ are locally Lipschitz and nondecreasing, and there exist constants $a_i > 0$ such that $\dot{\alpha}_i(x) \geq a_i$ for all $x \in \mathbb{R}$ at which α_i are differentiable, $i = 1, 2, \cdots, N$. Define matrix $A = \mathrm{diag}(a_1,\, a_2, \cdots, a_N)$.

(A_2) $f_i(x) \in \mathcal{IL}(\alpha),\ i = 1, 2, \cdots, N$.

3 Main Results

In this section, we will present our main results for neural networks (1) with (2).

Theorem 1. Under assumptions (A_1), (A_2), system (1) has one unique equilibrium point if there exist two matrices $R_{N\times N} > 0,\ Q_{N\times N} > 0$, a diagonal matrix $M = \mathrm{diag}(m_1, m_2, \cdots, m_N) > 0$ such that any one of the following matrix inequalities hold:

$$\text{(i)}\ \Xi_1 = R + c^2 Q + MD + (MD)^T + (MD^\tau) R^{-1} (MD^\tau)^T + (MB) Q^{-1} (MB)^T < 0, \tag{7}$$

$$\text{(ii)}\ \Xi_2 = \begin{pmatrix} -Q & (MB)^T & 0 \\ MB & R + c^2 Q + MD + (MD)^T & MD^\tau \\ 0 & (MD^\tau)^T & -R \end{pmatrix} < 0. \tag{8}$$

Proof. According to Schur complement, we can easily check that condition (i) is equivalent to condition (ii). In the following, we shall prove this theorem under condition (i) and in two steps.

Step I. In this step, the proof of existence for the equilibrium point will be given.

Note that the equilibrium point $(x^{*T}, S^{*T})^T$ of system (1) satisfies the following equations:

$$\begin{cases} -\alpha(x^*) + (D + D^\tau) f(x^*) + BS^* + I = 0, \\ -S^* + cf(x^*) = 0. \end{cases} \tag{9}$$

Equivalently, above equations can be rewritten as

$$\begin{cases} -\alpha(x^*) + (D + D^\tau + cB) f(x^*) + I = 0, \\ S^* = cf(x^*). \end{cases} \tag{10}$$

Now, define a map from $\mathbb{R}^N$ to $\mathbb{R}^N$ as :

$$\begin{aligned} F(x) &= \alpha(x) - (D + D^\tau + cB) f(x) - I \\ &= \widetilde{\alpha}(x) - (D + D^\tau + cB)\tilde{f}(x) + F(0), \end{aligned}$$

where $\widetilde{\alpha}(x) = \alpha(x) - \alpha(0)$, $\tilde{f}(x) = f(x) - f(0)$.

It follows from Lemma 3 that

$$\widetilde{\alpha}(x) = A^{(1)}(x)\, x, \ A^{(1)}(x) \in \bigcup_{z \in [0,x]} \partial\alpha(z),$$

where $A^{(1)}(x) = \mathrm{diag}(a_1^{(1)}(x), a_2^{(1)}(x), \cdots, a_N^{(1)}(x))$ with $a_i^{(1)}(x) \geq a_i, i = 1, 2, \cdots, N$.

Moreover, it is easy to see that $\tilde{f}_i \in \mathcal{IL}(\alpha)$ satisfying $\tilde{f}_i(0) = 0$ and $\tilde{f}_i(x_i)\, x_i > 0\, (x_i \neq 0)$. Let

$$\Omega = \{(x_1, x_2, \cdots, x_N)^T : |x_i| < R,\, i = 1, 2, \cdots, N.\}, \ R > 0,$$

and

$$\begin{aligned} H(\lambda, x) &= \widetilde{\alpha}(x) - \lambda\, (D + D^\tau + cB)\tilde{f}(x) + \lambda\, F(0) \\ &= A^{(1)}(x)\, x - \lambda\, (D + D^\tau + cB)\tilde{f}(x) + \lambda\, F(0), \end{aligned}$$

where $x \in \overline{\Omega} = \{(x_1, x_2, \cdots, x_N)^T : |x_i| \leq R,\, i = 1, 2, \cdots, N.\}$, $\lambda \in [0, 1]$.

By means of Lemma 4, we obtain that

$$\begin{aligned} \tilde{f}^T(x) M D^\tau \tilde{f}(x) &\leq \frac{1}{2} \left[\tilde{f}^T(x) (M D^\tau)\, R^{-1} (M D^\tau)^T \tilde{f}(x) + \tilde{f}^T(x)\, R\, \tilde{f}(x) \right], \\ c\tilde{f}^T(x) M B \tilde{f}(x) &\leq \frac{1}{2} \left[\tilde{f}^T(x) (MB)\, Q^{-1} (MB)^T \tilde{f}(x) + c^2 \tilde{f}^T(x)\, Q\, \tilde{f}(x) \right]. \end{aligned} \tag{11}$$

Applying (7) and (11), we can get

$$\begin{aligned}
&\tilde{f}^{\,T}(x)MH(\lambda,x)\\
&=\tilde{f}^{\,T}(x)M[A^{(1)}(x)\,x+\lambda\,F(0)]-\lambda\,\tilde{f}^{\,T}(x)M(D+D^{\tau}+cB)\tilde{f}(x)\\
&\geq\tilde{f}^{\,T}(x)M[A^{(1)}(x)\,x+\lambda\,F(0)]-\tfrac{\lambda}{2}\,\tilde{f}^{\,T}(x)\,\big[MD+(MD)^{T}\\
&\quad+(MD^{\tau})R^{-1}(MD^{\tau})^{T}+R+(MB)Q^{-1}(MB)^{T}+c^{2}Q\big]\,\tilde{f}(x)\\
&\geq\tilde{f}^{\,T}(x)M[A^{(1)}(x)\,x+\lambda\,F(0)]\\
&\geq\sum_{i=1}^{N}\Big[\Big|\tilde{f}_i(x_i)\Big|\,m_i\,a_i^{(1)}(x)\;|x_i|-\Big|\tilde{f}_i(x_i)\Big|\,m_i\,|F(0)_i|\Big]\\
&\geq\sum_{i=1}^{N}\Big|\tilde{f}_i(x_i)\Big|\,m_i\,a_i\left[|x_i|-\frac{|F(0)_i|}{a_i}\right],
\end{aligned}$$

where $F(0)_i\,(i=1,2,\cdots,N)$ denote the ith element of vector $F(0)$.

By virtue of Lemma 2, there exist constants $q_i>0$ and $r_i>0$, $i=1,2,\cdots,N$ such that

$$\Big|\tilde{f}_i(x_i)\Big|\geq q_i\,r_i^{\alpha},\quad\forall\;|x_i|\geq r_i,\;\;i=1,2,\cdots,N. \tag{12}$$

Let $r=\max\limits_{1\leq i\leq N}\{r_i\},\;a=\max\limits_{1\leq i\leq N}\dfrac{|F(0)_i|}{a_i}$, $\mathcal{N}_k=\{n_1,n_2,\cdots,n_k\}\subset\{1,2,\cdots,N\}$, $\forall\,k<N$. Define

$$\Omega_{\mathcal{N}_k}=\{x\in\mathbb{R}^k:\;|x_i|\leq a,\;i\in\mathcal{N}_k\}$$

and

$$\Phi_{\mathcal{N}_k}(x)=\sum_{i\in\mathcal{N}_k}m_i\,a_i\,|\tilde{f}_i(x_i)|\,[\,|x_i|-a].$$

Note that $\Omega_{\mathcal{N}_k}$ is a compact subset of $\mathbb{R}^k$, and $\Phi_{\mathcal{N}_k}(x)$ is continuous on $\Omega_{\mathcal{N}_k}$. Hence, $\Phi_{\mathcal{N}_k}(x)$ can reach its the minimum $\min\limits_{x\in\Omega_{\mathcal{N}_k}}\Phi_{\mathcal{N}_k}(x)$ on $\Omega_{\mathcal{N}_k}$.

Let $l=\min\limits_{1\leq i\leq N}\{m_i\,a_i q_i\,r_i^{\alpha}\}$, $\mathcal{M}_{\mathcal{N}_k}=\min\limits_{x\in\Omega_{\mathcal{N}_k}}\Phi_{\mathcal{N}_k}(x)$ and $\mathcal{M}=\min\{\mathcal{M}_{\mathcal{N}_k}:\mathcal{N}_k\subset\{1,2,\cdots,N\}\}$. Set $R>\max\left\{\sqrt{N}(a-\frac{\mathcal{M}}{l}),\sqrt{N}\,r\right\}$ and $x\in\partial\Omega$, then there exist two index sets $\mathcal{N}$ and $\overline{\mathcal{N}}$ such that

$$|x_i|\leq a,\;\;i\in\mathcal{N},\quad|x_i|>a,\;\;i\in\overline{\mathcal{N}},$$

where $\mathcal{N}\cup\overline{\mathcal{N}}=\{1,2,\cdots,N\}$. Furthermore, we can find an index $i_0\in\overline{\mathcal{N}}$ such that

$$|x_{i_0}|\geq\frac{R}{\sqrt{N}}\geq\max\{a,r\},\;\;|x_{i_0}|\geq\frac{R}{\sqrt{N}}>a-\frac{\mathcal{M}}{l}. \tag{13}$$

Note that $\mathcal{M} \leq 0$ and by using (12) and (13), for any $x \in \partial\Omega$ and $\lambda \in [0,1]$, we have

$$\begin{aligned}
&\tilde{f}^{\,T}(x) M H(\lambda, x) \\
&\geq \sum_{i\in\mathcal{N}} \left|\tilde{f}_i(x_i)\right| m_i\, a_i \left[|x_i| - \frac{|F(0)_i|}{a_i}\right] + \sum_{i\in\overline{\mathcal{N}}} \left|\tilde{f}_i(x_i)\right| m_i\, a_i \left[|x_i| - \frac{|F(0)_i|}{a_i}\right] \\
&\geq \mathcal{M} + \left|\tilde{f}_{i_0}(x_{i_0})\right| m_{i_0}\, a_{i_0} \left[|x_{i_0}| - \frac{|F(0)_{i_0}|}{a_{i_0}}\right] \\
&\geq \mathcal{M} + q_{i_0} r_{i_0}^{\alpha}\, m_{i_0}\, a_{i_0} [\,|x_{i_0}| - a] \\
&\geq q_{i_0} r_{i_0}^{\alpha}\, m_{i_0}\, a_{i_0} [\,|x_{i_0}| - a + \tfrac{\mathcal{M}}{l}] > 0.
\end{aligned}$$

Thus, we obtain that $H(\lambda, x) \neq 0$, $\forall\, x \in \partial\Omega$ and $\lambda \in [0,1]$. An application of Lemma 5 yields

$$\deg(H(0,x), \Omega, 0) = \deg(H(1,x), \Omega, 0),$$

i.e., $\deg(F(x), \Omega, 0) = \deg(A^{(1)}(x)\, x, \Omega, 0) = \operatorname{sgn}\left|A^{(1)}(x)\right| \neq 0$, where $\left|A^{(1)}(x)\right|$ denotes the determinant of matrix $A^{(1)}(x)$. It follows from Lemma 6 that $F(x) = 0$ has at least one solution in Ω. Thus, we obtain that model (1) has at least one equilibrium point $\left(x^{*T}, c\, f^{\,T}(x^*)\right)^T$.

Step II. In this step, we will prove the uniqueness of equilibrium point by the method of contradiction.

Assume that $\left(x_1^{*T}, S_1^{*T}\right)^T$ and $\left(x_2^{*T}, S_2^{*T}\right)^T$ are two different equilibrium points of system (1), then

$$\alpha(x_1^*) - \alpha(x_2^*) = (D + D^{\tau} + cB)\,(f(x_1^*) - f(x_2^*))\,. \tag{14}$$

From Lemma 3, we have that

$$\alpha(x_1^*) - \alpha(x_2^*) = A^{(2)}(x_1^* - x_2^*), \tag{15}$$

where $A^{(2)} \in \bigcup_{z\in[x_1^*, x_2^*]} \partial\alpha(z)$ and $A^{(2)} = \operatorname{diag}(a_1^{(2)}, a_2^{(2)}, \cdots, a_N^{(2)})$ with $a_i^{(2)} \geq a_i, i = 1, 2, \cdots, N$.

By means of Lemma 4, (7), (14) and (15), we can obtain

$$\begin{aligned}
0 &< (f(x_1^*) - f(x_2^*))^T\, M A^{(2)}\, (x_1^* - x_2^*) \\
&= (f(x_1^*) - f(x_2^*))^T\, M (D + D^{\tau} + cB)\, (f(x_1^*) - f(x_2^*)) \\
&\leq \frac{1}{2}\, (f(x_1^*) - f(x_2^*))^T \left[R + c^2 Q + MD + (MD)^T + (MD^{\tau}) R^{-1} (MD^{\tau})^T\right. \\
&\quad \left. + (MB) Q^{-1} (MB)^T\right] (f(x_1^*) - f(x_2^*)) < 0.
\end{aligned}$$

This is a contradiction. This completes the proof of the uniqueness of the equilibrium point for system (1).

Theorem 2. Under assumptions (A_1), (A_2) and $0 \leq \mu < 1$, system (1) has one unique equilibrium point which is globally exponentially stable if there exist two matrices $Q_{N\times N} > 0$, $R_{N\times N} > 0$, a positive definite diagonal matrix $M = \mathrm{diag}(m_1, m_2, \cdots, m_N) > 0$ such that the following linear matrix inequality holds:

$$\Xi_3 = \begin{pmatrix} -Q & (MB)^T & 0 \\ MB & R + c^2Q + MD + (MD)^T & MD^\tau \\ 0 & (MD^\tau)^T & -(1-\mu)\,R \end{pmatrix} < 0. \tag{16}$$

Proof. First of all, noticing the definition of negative definite matrix and $0 \leq \mu < 1$, it follows from $\Xi_3 < 0$ that $\Xi_2 < 0$. Thus, according to Theorem 1, the existence and uniqueness of equilibrium point for system (1) can be guaranteed under condition (16). In the following, we will prove that the condition (16) also ensures the global exponential stability of the unique equilibrium point via constructing a proper Lyapunov-Krasovskii functional.

Let $U_1(x(t), S(t)) = -\alpha(x(t)) + Df(x(t)) + D^\tau f(x(t-\tau(t))) + BS(t) + I$, $U_2(x(t), S(t)) = -S(t) + cf(x(t))$. Since $f_i \in \mathcal{IL}(\alpha)$ and therefore are continuous, $\alpha_i(x_i)$ are continuous (the definition of locally Lipschitz), $i = 1, 2, \cdots, N$, $U_1(x(t), S(t))$, $U_2(x(t), S(t))$ are continuous and local bounded. Hence, we can obtain the existence of the local solution for system (1) with initial values $x(t) = \phi(t)$, $S(t) = \varphi(t)$, $t \in [-\tau, 0]$ on $[0, t^*)$, where $t^* \in (0, +\infty)$ or $t^* = +\infty$, and $[0, t^*)$ is the maximal right-side existence interval of the local solution.

Let $({x^*}^T, {S^*}^T)^T$ be the unique equilibrium point of system (1). Make a transformation $u(t) = x(t) - x^*$, $v(t) = S(t) - S^*$, then system (1) is transformed into

$$\begin{cases} \dot{u}(t) = -\bar{\alpha}(u(t)) + D\bar{f}(u(t)) + D^\tau \bar{f}\,(u(t-\tau(t))) + Bv(t), \\ \dot{v}(t) = -v(t) + c\bar{f}(u(t)), \end{cases} \tag{17}$$

where $\bar{\alpha}(u(t)) = \alpha(x^* + u(t)) - \alpha(x^*)$, $\bar{f}(u(t)) = f(x^* + u(t)) - f(x^*)$.

Similarly to (15), from Lemma 3, we have

$$\bar{\alpha}(u(t)) = A^{(3)}(t)\,u(t), \quad A^{(3)}(t) \in \bigcup_{z\in[x^*,\ x^*+u(t)]} \partial\alpha(z), \tag{18}$$

where $A^{(3)}(t) = \mathrm{diag}(a_1^{(3)}(t), a_2^{(3)}(t), \cdots, a_N^{(3)}(t))$ with $a_i^{(3)}(t) \geq a_i$, $i = 1, 2, \cdots, N$.

From (16), we can choose a small constant $0 < \gamma < \min\{a_i,\ i = 1, 2, \cdots, N\}$ such that

$$\Xi_4 = \begin{pmatrix} (\gamma - 1)\,Q & (MB)^T & 0 \\ MB & e^{\gamma\tau}R + c^2Q + MD + (MD)^T & MD^\tau \\ 0 & (MD^\tau)^T & -(1-\mu)\,R \end{pmatrix} < 0. \tag{19}$$

Consider the following Lyapunov-Krasovskii functional:

$$V(t) = V_1(t) + V_2(t) + V_3(t),$$

where

$$\begin{aligned} V_1(t) &= e^{\gamma t}\, v^T(t)\, Q\, v(t), \\ V_2(t) &= 2\, e^{\gamma t} \sum_{i=1}^{N} m_i \int_0^{u_i(t)} \bar{f}_i(s)\, \mathrm{d}s, \\ V_3(t) &= \int_{t-\tau(t)}^{t} \bar{f}^T(u(s))\, R\, \bar{f}(u(s))\, e^{\gamma(s+\tau)} \mathrm{d}s. \end{aligned} \tag{20}$$

Calculating the time derivative of $V_i (i = 1, 2, 3)$ along the trajectories of system (17) on $[0, t^*)$ gives

$$\begin{aligned} \dot{V}_1(t) &= \gamma\, e^{\gamma t}\, v^T(t)\, Q\, v(t) + 2\, e^{\gamma t}\, v^T(t)\, Q\, \left[-v(t) + c\bar{f}(u(t))\right] \\ &= e^{\gamma t} \left[(\gamma - 2)\, v^T(t)\, Q\, v(t) + 2\, c\, v^T(t)\, Q\, \bar{f}(u(t))\right]. \end{aligned} \tag{21}$$

Using Lemma 4, we obtain that

$$\begin{aligned} 2\, v^T(t)\, Q\, \left(c\bar{f}(u(t))\right) &\le v^T(t)\, Q\, Q^{-1}\, Q^T\, v(t) + c^2\, \bar{f}^T(u(t))\, Q\, \bar{f}(u(t)) \\ &= v^T(t)\, Q\, v(t) + c^2\, \bar{f}^T(u(t))\, Q\, \bar{f}(u(t)). \end{aligned} \tag{22}$$

Substituting (22) into (21) results in

$$\dot{V}_1(t) \le e^{\gamma t} \left[(\gamma - 1)\, v^T(t)\, Q\, v(t) + c^2\, \bar{f}^T(u(t))\, Q\, \bar{f}(u(t))\right]. \tag{23}$$

$$\begin{aligned} \dot{V}_2(t) &= e^{\gamma t} \left[2\, \gamma \sum_{i=1}^{N} m_i \int_0^{u_i(t)} \bar{f}_i(s)\, \mathrm{d}s + 2 \sum_{i=1}^{N} m_i \bar{f}_i(u_i(t))\, \dot{u}_i(t)\right] \\ &\le e^{\gamma t} \left\{2\, \gamma\, u^T(t) M \bar{f}(u(t)) + 2\, \bar{f}^T(u(t)) M[-A^{(3)}(t)\, u(t)\right. \\ &\quad \left. + D\bar{f}(u(t)) + D^\tau \bar{f}\,(u(t - \tau(t))) + Bv(t)]\right\} \\ &\le e^{\gamma t} \left[2\, u^T(t)(\gamma M - AM)\bar{f}(u(t)) + 2\, \bar{f}^T(u(t)) MD\bar{f}(u(t))\right. \\ &\quad \left. + 2\, \bar{f}^T(u(t)) MD^\tau \bar{f}(u(t - \tau(t))) + 2\, \bar{f}^T(u(t)) MB\, v(t)\right] \\ &\le e^{\gamma t}[2\, \bar{f}^T(u(t)) MD\bar{f}(u(t)) + 2\, \bar{f}^T(u(t)) MD^\tau \bar{f}(u(t - \tau(t))) \\ &\quad + 2\, \bar{f}^T(u(t)) MB\, v(t)]. \end{aligned} \tag{24}$$

Herein, in the last inequality of (24), the inequality $0 < \gamma < \min\{a_i, i = 1, 2, \cdots, N\}$ has been applied.

$$\dot{V}_3(t) \le e^{\gamma t} \left[e^{\gamma\, \tau}\, \bar{f}^T(u(t))\, R\, \bar{f}(u(t)) - (1 - \mu)\, \bar{f}^T(u(t - \tau(t)))\, R\, \bar{f}(u(t - \tau(t)))\right]. \tag{25}$$

Let $\xi(t) = \begin{pmatrix} v(t) \\ \bar{f}(u(t)) \\ \bar{f}(u(t - \tau(t))) \end{pmatrix}$, and then from (23)-(25), we get

$$\dot{V}(t) \le e^{\gamma t}\, \xi^T(t)\, \Xi_4\, \xi(t). \tag{26}$$

This implies $\dot{V}(t) < 0$ for any $\xi^T(t) \neq 0$, which gives $V(t) \leq V(0),\ t \in [0, t^*)$. Furthermore, it follows from (20) that

$$\lambda_{\min}(Q)\, v^T(t)\, v(t) \leq v^T(t)\, Q\, v(t) \leq V(0)\, e^{-\gamma t} \leq V(0), \tag{27}$$

and

$$2 \sum_{i=1}^{N} m_i \int_0^{u_i(t)} \bar{f}_i(s)\, \mathrm{d}s \leq V(0)\, e^{-\gamma t} \leq V(0). \tag{28}$$

Obviously, it follows from (27) that $v_i(t)$ are bounded on $[0, t^*)$. Moreover, according to (28) and Lemma 1, we can derive that $u_i(t)$ are bounded on $[0, t^*)$. Thus, by virtue of the continuous theorem[12], we conclude that $t^* = +\infty$.

Since $\bar{f}_i(s) \in \mathcal{IL}(\alpha),\ \bar{f}_i(0) = 0$, by Lemma 2, there exist constants $q_i > 0$ and $r_i > 0$, $i = 1, 2, \cdots, N$ such that

$$|\bar{f}_i(s)| \geq q_i |s|^\alpha, \quad \forall\ |s| \leq r_i. \tag{29}$$

Moreover, from (28), we can obtain that

$$\lim_{t \to +\infty} u_i(t) = 0, \quad i = 1, 2, \cdots, N.$$

Thus, there exists a constant $T > 0$ such that $u_i(t) \in [-r_0, r_0],\ t \geq T$ hold for all $i = 1, 2, \cdots, N$, where $r_0 = \min\limits_{1 \leq i \leq N} \{r_i\}$, r_i are the constants in condition (29).

Let $m = \min\{m_i,\ i = 1, 2, \cdots, N\},\ q = \min\{q_i,\ i = 1, 2, \cdots, N\}$, from (28) and (29), we have

$$\begin{aligned} \frac{V(0)}{2}\, e^{-\gamma t} &\geq \sum_{i=1}^{N} m_i \int_0^{u_i(t)} \bar{f}_i(s) \mathrm{d}s \geq \sum_{i=1}^{N} m_i \int_0^{|u_i(t)|} q_i |s|^\alpha \mathrm{d}s \\ &\geq \frac{m\, q}{\alpha + 1} \left\{ \max_{1 \leq i \leq N} |u_i(t)| \right\}^{\alpha+1}, \quad t \geq T. \end{aligned}$$

That is,

$$\max_{1 \leq i \leq N} |u_i(t)| \leq \left[\frac{V(0)(1+\alpha)}{2\, m\, q} \right]^{\frac{1}{1+\alpha}} e^{-\frac{\gamma}{1+\alpha} t}, \quad t \geq T. \tag{30}$$

Let $\gamma' = \min\{\frac{\gamma}{2},\ \frac{\gamma}{1+\alpha}\}$, $M = \sqrt{\frac{V(0)}{\lambda_{\min}(Q)}} + \sqrt{N} \left(\frac{V(0)(1+\alpha)}{2\, m\, q} \right)^{\frac{1}{1+\alpha}}$, from (27) and (30), we get that

$$\|x(t) - x^*\| + \|S(t) - S^*\| \leq M e^{-\gamma' t},$$

for all $t > T$. This completes the proof of Theorem 2.

4 Conclusions

In this paper, a novel class of competitive neural networks has been presented with inverse Lipschitz neuron activations. By employing nonsmooth analysis method, applying LMI technique, topological degree theory and Lyapunov-Kras ovskii function approach, we proved the existence, uniqueness and global exponential stability of the equilibrium point for competitive neural networks with time-varying delay.

Acknowledgements. This work was jointly supported by the National Natural Science Foundation of China under Grant 60874088, the 333 Project of Jiangsu Province of China, and the Specialized Research Fund for the Doctoral Program of Higher Education under Grant 20070286003.

References

1. Meyer-Bäse, A., Pilyugin, S., Chen, Y.: Global Exponential Stability of Competitive Neural Networks with Different Time Scales. IEEE Trans. Neural Networks 14, 716–719 (2003)
2. Meyer-Bäse, A., Thümmler, V.: Local and Global Stability of An Unsupervised Competitive Neural Network. IEEE Trans. Neural Networks 19, 346–351 (2008)
3. Lu, H., Shun-ichi, A.: Global Exponential Stability of Multitime Scale Competitive Neural Networks with Nonsmooth Functions. IEEE Trans. Neural Networks 17, 1152–1164 (2006)
4. Lu, H., He, Z.: Global Exponential Stability of Delayed Competitive Neural Networks with Different Time Scales. Neural Networks 18, 243–250 (2005)
5. Nie, X., Cao, J.: Exponential Stability of Competitive Neural Networks with Time-Varying and Distributed Delays. Proc. IMechE, Part I: Journal of Systems and Control Engineering 222, 583–594 (2008)
6. Nie, X., Cao, J.: Multistability of Competitive Neural Networks with Time-Varying and Distributed Delays. Nonlinear analysis: Real World Applications 10, 928–942 (2009)
7. Wu, H., Xue, X.: Stability Analysis for Neural Networks with Inverse Lipschitzian Neuron Activations and Impulses. Appl. Math. Modelling 32, 2347–2359 (2008)
8. Wu, H.: Global Exponential Stability of Hopfield Neural Networks with Delays and Inverse Lipschitz Neuron Activations. Nonlinear analysis: Real World Applications 10, 2297–2306 (2009)
9. Nie, X., Cao, J.: Stability Analysis for the Generalized Cohen-Grossberg Neural Networks with Inverse Lipschitz Neuron Activations. Computers and Mathematics with Applications 57, 1522–1536 (2009)
10. Yu, W., Cao, J., Wang, J.: An LMI Approach to Global Asymptotic Stability of the Delayed Cohen-Grossberg Neural Network via Nonsmooth Analysis. Neural Networks 20, 810–818 (2007)
11. Deimling, K.: Nonlinear Functional Analysis. Springer, Berlin (1985)
12. Miller, P., Michel, A.: Differential Equations. Academic, New York (1982)

Stability and Hopf Bifurcation of a BAM Neural Network with Delayed Self-feedback

Shifang Kuang[1], Feiqi Deng[1], and Xuemei Li[2]

[1] College of Automation Science and Engineering, South China University of Technology, Guangzhou 510640, China
[2] College of Mathematics and Computer, Hunan Normal University, Changsha 410081, China

Abstract. In this paper, we consider a bidirectional associate memory(BAM) neural networks with delayed self-feedback. Regarding the self-connection delay as the bifurcation parameter, the linear stability and Hopf bifurcation analysis are carried out. The stability and direction of the Hopf bifurcation are determined by applying the normal form theory and the center manifold reduction. Numerical simulation results are given to support the theoretical predictions.

Keywords: Neural network, Delay, Stability, Hopf bifurcation.

1 Introduction

Since Hopfied[1] constructed a simplified neural network model in 1984, dynamical characteristics of neural networks have become a subject of intensive research activity. Based on the Hopfied neural network model, Marcus and Westervelt[2] proposed a neural network model incorporating time delay. Afterward, a variety of artificial models has been established to describe neural networks with delays. In [3,4], a class of two-layer heteroassociative networks, called bidirectional associative memory(BAM) neural networks with or without axonal signal transmission delays, has been proposed and applied in many fields such as pattern recognition and automatic control.

The delayed BAM neural network is described by the following system

$$\begin{cases} \dot{x}_i(t) = -\mu_i x_i(t) + \sum_{j=1}^{m} c_{ji} f_i(y_j(t-\tau_{ji})) + I_i, \\ \dot{y}_j(t) = -\nu_j y_j(t) + \sum_{i=1}^{n} d_{ij} g_j(x_i(t-\gamma_{ij})) + J_j, \end{cases} \tag{1}$$

where, $d_{ij}, c_{ji}(i = 1, 2, \cdots, n, j = 1, 2, \cdots, m)$ are weights of connection through the neurons in two layers: the I-layer and the J-layer; μ_i and ν_j describe the stability of internal neurons processes on the I-layer and the J-layer, respectively. On the I-layer, the neurons whose states are denoted by $x_i(t)$ receive the inputs I_i and the inputs outputted by those neurons in the J-layer via activation functions f_i, while on the J-layer, the neurons whose states are denoted by $y_j(t)$ receive the inputs J_j and the inputs outputted by those neurons in the I-layer via activation functions g_j. Parameters τ_{ji} and γ_{ij} correspond to the finite time delays of neural processing and delivery of signals.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 493–503, 2010.

In [3], Wang and Zou considered a special case of (1) when all delays in each layer are identical. In [4], Song et al. assumed that time delay from the I-layer to another J-layer is τ_1, while the time delay from the J-layer back to the I-layer is τ_2, and there are only one neuron in the I-layer and two neurons in the J-layer. Thus, the simplified BAM neural network model takes the form

$$\begin{cases} \dot{x}_1(t) = -\mu_1 x_1(t) + c_{21} f_1(y_1(t-\tau_2)) + c_{31} f_1(y_2(t-\tau_2)), \\ \dot{y}_1(t) = -\mu_2 y_1(t) + c_{12} f_2(x_1(t-\tau_1)), \\ \dot{y}_2(t) = -\mu_3 y_2(t) + c_{13} f_3(x_1(t-\tau_1)). \end{cases} \tag{2}$$

On the other hand, recent work [5,6] has shown that inhibitory self-connections play a role in stabilizing a network under some conditions on delays. This motivates us to incorporate inhibitory self-connections terms into the model system (2). For the sake of simplicity, we shall study a BAM neural network modeled by the following nonlinear differential system

$$\begin{cases} \dot{x}_1(t) = -kx_1(t) + Mf(x_1(t-\sigma)) + c_{21} f_1(y_1(t-\tau_2)) + c_{31} f_1(y_2(t-\tau_2)), \\ \dot{y}_1(t) = -ky_1(t) + Mf(y_1(t-\sigma)) + c_{12} f_2(x_1(t-\tau_1)), \\ \dot{y}_2(t) = -ky_2(t) + Mf(y_2(t-\sigma)) + c_{13} f_3(x_1(t-\tau_1)), \end{cases} \tag{3}$$

where $k > 0, c_{1j} (j = 2, 3)$ and $c_{i1} (i = 2, 3)$ are real constants. Our aim in this paper is to study the stability of the zero solution of (3) and Hopf bifurcations. Taking the identical delay as a parameter, we shall show that when the delay σ passes through a critical value, the zero solution loses its stability and a Hopf bifurcation occurs.

2 Stability and Local Hopf Bifurcation

Throughout this paper, we always assume that $\tau_2 > \tau_1 > 0$ and $f_i \in c^1$, $f_i(0) = 0$, for $i = 1, 2, 3$. Linearizing (3) at the origin leads to

$$\begin{cases} \dot{x}_1(t) = -kx_1(t) + \beta x_1(t-\sigma) + \alpha_{21} y_1(t-\tau_2) + \alpha_{31} y_2(t-\tau_2), \\ \dot{y}_1(t) = -ky_1(t) + \beta y_1(t-\sigma) + \alpha_{12} x_1(t-\tau_1), \\ \dot{y}_2(t) = -ky_2(t) + \beta y_2(t-\sigma) + \alpha_{13} x_1(t-\tau_1), \end{cases} \tag{4}$$

where $\alpha_{ij} = c_{ij} f_j'(0)$ and $\beta = Mf'(0)$. Let $\tau = \frac{\tau_1 + \tau_2}{2}$. The associated characteristic equation of (4) is

$$(\lambda + k - \beta e^{-\lambda\sigma})[(\lambda + k - \beta e^{-\lambda\sigma})^2 - \alpha e^{-2\lambda\tau}] = 0 \quad (\alpha = \alpha_{12}\alpha_{21} + \alpha_{13}\alpha_{31}) \tag{5}$$

Throughout this paper, we assume that $\alpha > 0$. So either

$$\lambda + k - \beta e^{-\lambda\sigma} = 0, \tag{6}$$

or

$$\lambda + k - \beta e^{-\lambda\sigma} \pm \sqrt{\alpha} e^{-\lambda\tau} = 0. \tag{7}$$

Substituting $\lambda = \mu + i\omega(\omega \geq 0)$ into the left sides of both (6) and (7) and separating the real and imaginary parts, we obtain

$$\begin{cases} Re_{(6)}(\mu,\omega) = \mu + k - \beta e^{-\mu\sigma}\cos(\omega\sigma) = 0, \\ Im_{(6)}(\mu,\omega) = \omega + \beta e^{-\mu\sigma}\sin(\omega\sigma) \quad = 0. \end{cases}$$

$$\begin{cases} Re_{(7)}(\mu,\omega) = \mu + k - \beta e^{-\mu\sigma}\cos(\omega\sigma) \pm \sqrt{\alpha}e^{-\mu\tau}\cos(\omega\tau) = 0, \\ Im_{(7)}(\mu,\omega) = \omega + \beta e^{-\mu\sigma}\sin(\omega\sigma) \mp \sqrt{\alpha}e^{-\mu\tau}\sin(\omega\tau) \quad = 0. \end{cases}$$

Theorem 1. *If $k > |\beta| + \sqrt{\alpha}, \sigma \geq 0, \tau \geq 0$, then all roots of (5) have negative real parts, and hence the trivial solution of (3) is asymptotically stable.*

Proof. Notice that

$$Re_{(6)}(\mu,\omega) \geq k - |\beta|, \qquad Re_{(7)}(\mu,\omega) \geq k - |\beta| - \sqrt{\alpha}$$

for all $\mu \geq 0, \sigma \geq 0, \tau \geq 0$. Obviously, if $k > |\beta|$, then the root of (6) has negative real part, and if $k > |\beta| + \sqrt{\alpha}$, then the roots of (7) have negative parts. Hence, if $k > |\beta| + \sqrt{\alpha}$, then all roots of (5) have negative real parts. This completes the proof. □

Theorem 2. *If $\beta < 0$, and $\sqrt{\alpha} < -\beta, \sigma \in [0, \frac{1}{-2\beta}]$, then for all $\tau \geq 0$, the trivial solution of (3) is asymptotically stable.*

Proof. From $Re_{(7)}(\mu,\omega) = 0$ and $Im_{(7)}(\mu,\omega) = 0$, we obtain

$$\mu + k - \beta e^{-\mu\sigma}\cos(\omega\sigma) = \mp\sqrt{\alpha}e^{-\mu\tau}\cos(\omega\tau),$$
$$\omega + \beta e^{-\mu\sigma}\sin(\omega\sigma) = \pm\sqrt{\alpha}e^{-\mu\tau}\sin(\omega\tau),$$

and hence

$$(\mu+k)^2 + \omega^2 - 2\beta e^{-\mu\sigma}[(\mu+k)\cos(\omega\sigma) - \omega\sin(\omega\sigma)] + \beta^2 e^{-2\mu\sigma} - \alpha e^{-2\mu\tau} = 0. \tag{8}$$

If we assume that

$$\beta < 0, \text{and} \sqrt{\alpha} < -\beta, \sigma \in [0, \frac{1}{-2\beta}],$$

it follows from $Im_{(7)}(\mu,\omega) = 0$ that

$$\omega < -2\beta \text{ for } \mu \geq 0, \tau \geq 0 \text{ and } \omega\sigma \in [0,1].$$

Letting the left-hand side of (8) be $M(\mu,\omega)$, for fixed ω, we have

$$\begin{aligned} M(0,\omega) &= k^2 + \beta^2 - \alpha - 2\beta k\cos(\omega\sigma) + \omega^2 + 2\beta\omega\sin(\omega\sigma) \\ &> \omega^2(1 + 2\beta\sigma) \\ &\geq 0(\text{because} - 1 \leq 2\beta\sigma \leq 0). \end{aligned}$$

Differentiating $M(\mu,\omega)$ with respect to μ, we have

$$\begin{aligned} \frac{\partial M(\mu,\omega)}{\partial\mu} &= 2\{(\mu+k)[1 + \beta\sigma e^{-\mu\sigma}\cos(\omega\sigma)] - \beta e^{-\mu\sigma}[\cos(\omega\sigma) + \beta\sigma e^{-\mu\sigma}] \\ &\quad + \alpha\tau e^{-2\mu\tau} - \beta\sigma\omega e^{-\mu\sigma}\sin(\omega\sigma)\}. \end{aligned}$$

Because $\beta < 0, \sigma \in [0, \frac{1}{-2\beta}]$, then $-\beta\omega\sigma e^{-\mu\sigma}\sin(\omega\sigma)$ is nonnegative, meanwhile $\alpha\tau e^{-2\mu\tau}$ is also nonnegative. Another $\omega\sigma \in [0,1]$ and $\frac{1}{2} = \cos(\pi/3) < \cos 1 \leq \cos(\omega\sigma) \leq 1$,we have

$$(\mu + k)[1 + \beta\sigma e^{-\mu\sigma}\cos(\omega\sigma)] \geq (\mu + k)(1 - \frac{1}{2}) > 0$$

and

$$-\beta e^{-\mu\sigma}[\cos(\omega\sigma) + \beta\sigma e^{-\mu\sigma}] \geq -\beta e^{-\mu\sigma}(\cos 1 - \frac{1}{2}) > 0.$$

Therefore, $\partial M(\mu,\omega)/\partial\mu > 0$. From above discussion, we have $M(\mu,\omega) > 0$ for $\mu \geq 0$, which contradicts with (8).

From $Re_{(8)}(\mu,\omega) = 0$, we can easily obtain the root of (8) has negative real part, if $\beta < 0, \sigma \in [0, \frac{1}{-2\beta}]$. This completes the proof. □

In the following, we will regard σ as the parameter and try to find its critical value at which the bifurcation occurs. Letting $\sigma = 0$ in (5). We have

$$Re_{(6)}(\mu,\omega) = \mu + k - \beta \geq k - \beta,$$

$$Re_{(7)}(\mu,\omega) = \mu + k - \beta \pm \sqrt{\alpha}e^{-\mu\tau}\cos(\omega\tau) \geq k - \beta - \sqrt{\alpha},$$

for all $\mu \geq 0$, from which, we obtain the following result.

Lemma 1. *If the coefficients satisfy*

$$\beta < k - \sqrt{\alpha}, \tag{9}$$

then all roots of (5) have negative real parts at $\sigma = 0$ for all $\tau \geq 0$.

Next we investigate if $\sigma \geq 0$ will destroy the stability. Theorem 1 and Lemma 1 suggest that in order to explore the possibility that $\sigma \geq 0$ destroys the stability, we need to assume that (9) and $|\beta| + \sqrt{\alpha} \geq k$ hold, or equivalently,

$$\beta < -|k - \sqrt{\alpha}|. \tag{10}$$

Under this assumption, we know for any fixed $\tau \geq 0$, all roots of (5) have negative real parts when $\sigma = o$ and it is possible for some roots having nonnegative real parts when $\sigma > 0$.

Note that $\lambda = 0$ cannot be a root of (5) due to (10), and $\lambda = i\omega$ with $\omega > 0$ is a root of (5) if and only if

$$\begin{cases} k - \beta\cos(\omega\sigma) = 0, \\ \omega + \beta\sin(\omega\sigma) = 0, \end{cases} \text{or} \begin{cases} \beta\cos(\omega\sigma) = k \pm \sqrt{\alpha}\cos(\omega\tau), \\ \beta\sin(\omega\sigma) = -\omega \pm \sqrt{\alpha}\sin(\omega\tau). \end{cases} \tag{11}$$

From (11) we have

$$\omega_1 = \sqrt{\beta^2 - k^2},$$

$$\text{or } \beta^2 = k^2 + \omega^2 + \alpha \pm 2\sqrt{\alpha}[k\cos(\omega\tau) - \omega\sin(\omega\tau)]. \tag{12}$$

(12) can have either finite number of roots or no root for $\omega > 0$. In the case of finite number of roots, we denote them by ω_k, $k = 2, 3$. It follows from (11) that

$$\sigma_{1i} = \frac{1}{\omega_1}(\arccos\frac{k}{\beta} + 2i\pi),\ i = 0, 1, 2, \cdots$$

$$\sigma_{kj}^{\pm} = \frac{1}{\omega_k}(\arccos\frac{k \pm \sqrt{\alpha}\cos(\omega_k\tau)}{\beta} + 2j\pi), j = 0, 1, 2, \cdots \tag{13}$$

In the case where (12) has no root, we denote the corresponding $\sigma_{kj}(0) = \infty$.

Define

$$\sigma_0 = \sigma_{k0} = \min\{\sigma_{k0}^{\pm}(0), \sigma_{10}\}, \qquad \omega_0 = \omega_{k0}.$$

The above analysis and a direct calculation give

Lemma 2. *Assume that (10) holds. Then*

(i) all roots of (5) have negative real parts for any fixed $\tau \geq 0$ and for $\sigma \in [0, \sigma_0)$;
(ii) (5) has a pair of simple purely imaginary roots and all other roots have negative real parts at $\sigma = \sigma_0$;
(iii) at least one root of (5) has positive real part if $\sigma > \sigma_0$.

Remark 1. Let $\lambda(\sigma) = \mu(\sigma) + i\omega(\sigma)$ be the root of (5) near $\sigma = \sigma_{kj}$ satisfying

$$\mu(\sigma_{kj}) = 0, \qquad \omega(\sigma_{kj}) = \omega_k. \tag{14}$$

Substituting $\lambda(\sigma)$ into (5) and taking the derivative with respect to σ, we obtain

$$\begin{aligned}&[(\lambda + k - \beta e^{-\lambda\sigma})^2 - \alpha e^{-2\lambda\tau}][(1 + \beta\sigma e^{-\lambda\sigma}\tfrac{d\lambda}{d\sigma})]\\ &\quad +(\lambda + k - \beta e^{-\lambda\sigma})\{2(\lambda + k - \beta e^{-\lambda\sigma})[(1 + \beta\sigma e^{-\lambda\sigma}\tfrac{d\lambda}{d\sigma})] + 2\tau\alpha e^{-2\lambda\tau}\tfrac{d\lambda}{d\sigma}\} = 0.\end{aligned}$$

Then

$$\left(\frac{d\lambda}{d\sigma}\right)^{-1} = \frac{(1+\beta\sigma e^{-\lambda\sigma})[3(\lambda+k-\beta e^{-\lambda\sigma})^2-\alpha e^{-2\lambda\tau}]+2(\lambda+k-\beta e^{-\lambda\sigma})\tau\alpha e^{-2\lambda\tau}}{\lambda\beta e^{-\lambda\sigma}[\alpha e^{-2\lambda\tau} - 3(\lambda + k - \beta e^{-\lambda\sigma})^2]}. \tag{15}$$

From (14) and (15), we can obtain

$$\begin{aligned}Re\left[\tfrac{d\lambda}{d\sigma}\right]^{-1}_{\sigma=\sigma_{kj}} &= \tfrac{1}{M^2+N^2}\{M[(-3\omega_k^2 + 3k^2 + 3\beta^2 - 6\beta^2 k\sigma_{kj})\cos(\omega_k\sigma_{kj}) - 6\omega_k(k+\\ &\quad \beta^2\sigma_{kj})\sin(\omega_k\sigma_{kj}) + 3\beta(k^2\sigma_{kj} - 2k - \sigma_{kj}\omega^2) + 3\beta^2\sigma_{kj}\cos(2\omega_k\sigma_{kj})\\ &\quad +(\alpha + 2k\tau\alpha)\cos(\omega_k\sigma_{kj} - 2\omega_k\tau) + (\alpha\beta\sigma_{kj} - 2\alpha\beta\tau)\cos(2\omega_k\tau) -\\ &\quad 2\omega_k\tau\alpha\sin(\omega_k\sigma_{kj} - 2\omega_k\tau)] + N[(-3\omega_k^2 + 3k^2 - 3\beta^2 - 6k\beta^2\sigma_{kj})\times\\ &\quad \sin(\omega_k\sigma_{kj}) + 6\omega_k(k + \beta^2\sigma_{kj})\cos(\omega_k\sigma_{kj}) + 6\omega_k\beta(k\sigma_{kj} - 1)\\ &\quad -3\beta^3\sigma_{kj}\sin(2\omega_k\sigma_{kj}) + (\alpha + 2k\tau\alpha)\sin(\omega_k\sigma_{kj} - 2\omega_k\tau)\\ &\quad +(2\alpha\beta\tau - \alpha\beta\sigma_{kj})\sin(2\omega_k\tau) + 2\omega_k\tau\alpha\cos(\omega_k\sigma_{kj} - 2\omega_k\tau)]\},\end{aligned}$$

where

$$\begin{aligned}M &= \alpha\beta\omega_k\sin(2\omega_k\tau) - 3\beta^3\omega_k\sin(2\omega_k\sigma_{kj}) - 2\omega_k^2\beta k\\ &\quad +2\beta^2\omega_k^2\cos(\omega_k\sigma_{kj}) - 2k\beta^2\omega_k\sin(\omega_k\sigma_{kj}),\\ N &= \alpha\beta\omega_k\cos(2\omega_k\tau) + 3\omega_k^3\beta - 3k^2\omega_k\beta - 3\beta^3\omega_k\cos(2\omega_k\sigma_{kj})\\ &\quad -2\beta^2\omega_k^2\sin(\omega_k\sigma_{kj}) - 2k\beta^2\omega_k\cos(\omega_k\sigma_{kj}).\end{aligned}$$

Theorem 3. *Based on the lemmas presented in the above, we have the following results.*

(i) If $\beta < k - \sqrt{\alpha}$, *then the trivial solution of (3) is asymptotically stable at* $\sigma = 0$ *for all* $\tau \geq 0$.
(ii) If $\beta < -|k - \sqrt{\alpha}|$, *then the trivial solution of (3) is asymptotically stable for* $\sigma \in [0, \sigma_0)$ *and unstable of* $\sigma > \sigma_0$.
(iii) Hopf bifurcation occurs at $\sigma = \sigma_0$ *if (10) holds and* $Re\left[\frac{d\lambda}{d\sigma}\right]_{\sigma=\sigma_{kj}} \neq 0$.

Remark 2. Note that the stability analysis and Hopf bifurcation on system (3) is under the condition $\alpha > 0$. In fact, when $\alpha \leq 0$, our method is also effective.

3 Direction and Stability of the Hopf Bifurcating Periodic Solution

In this section, we discuss the direction of Hopf bifurcation and the stability of the bifurcating periodic solution. First, we rewrite the DDE(3) as an ODE by using the Riesz representation theorem. Then we use the method of Hassard et al.[7] to establish the results on Hopf bifurcation.

Letting $u(t) = (x_1(t), y_1(t), y_2(t))^T$ and $u_t(\theta) = u(t + \theta)$ for $\theta \in [-\tau_2, 0)$, we can rewrite (3) as

$$\dot{u}(t) = L_\sigma(u_t) + F(u_t, \sigma), \tag{16}$$

with

$$L_\sigma(\phi) = -K\phi(0) + B_1\phi(-\sigma) + B_2\phi(-\tau_2) + B\phi(-\tau_1), \tag{17}$$

where

$$K = \begin{pmatrix} k & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & k \end{pmatrix}, B_1 = \begin{pmatrix} \beta & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & \beta \end{pmatrix}, B_2 = \begin{pmatrix} 0 & \alpha_{21} & \alpha_{31} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, B = \begin{pmatrix} 0 & 0 & 0 \\ \alpha_{12} & 0 & 0 \\ \alpha_{13} & 0 & 0 \end{pmatrix},$$

and

$$F(\phi, \sigma) = \begin{pmatrix} l_1\phi_1^2(-\sigma) + l_2\phi_1^3(-\sigma) + l_3\phi_2^2(-\tau_2) + l_4\phi_2^3(-\tau_2) \\ \qquad + l_5\phi_3^2(-\tau_2) + l_6\phi_3^3(-\tau_2) + \cdots \\ m_1\phi_1^2(-\tau_1) + m_2\phi_1^3(-\tau_1) + l_1\phi_2^2(-\sigma) + l_2\phi_2^3(-\sigma) + \cdots \\ n_1\phi_1^2(-\tau_1) + n_2\phi_1^3(-\tau_1) + l_1\phi_3^2(-\sigma) + l_2\phi_3^3(-\sigma) + \cdots \end{pmatrix}, \tag{18}$$

where, $\phi(\theta) = (\phi_1(\theta), \phi_2(\theta), \phi_3(\theta))^T \in C([-\tau_2, 0], R^3), l_1 = \frac{Mf_1''(0)}{2}, l_2 = \frac{Mf_1'''(0)}{6}$, $l_3 = \frac{c_{21}f_1''(0)}{2}, l_4 = \frac{c_{21}f_1'''(0)}{6}, l_5 = \frac{c_{31}f_1''(0)}{2}, l_6 = \frac{c_{31}f_1'''(0)}{6}, m_1 = \frac{c_{12}f_2''(0)}{2}, m_2 = \frac{c_{12}f_2'''(0)}{6}, n_1 = \frac{c_{13}f_3''(0)}{2}, n_2 = \frac{c_{13}f_3'''(0)}{6}$.

By the Riesz representation theorem, there exists a bounded variation function matrix $\eta(\theta, \sigma)$ for $\theta \in [-\tau_2, 0]$ such that

$$L_\sigma\phi = \int_{-\tau_2}^{0} d\eta(\theta, \sigma)\phi(\theta), \ \ for\ \phi \in C([-\tau_2, 0], R^3). \tag{19}$$

In fact, we can choose

$$\eta(\theta,\sigma)=\begin{cases}-kId_3, & \theta=0,\\ B_1\delta(\theta+\sigma), & \theta\in[-\sigma,0),\\ B\delta(\theta+\tau_1), & \theta\in[-\tau_1,-\sigma),\\ B_2\delta(\theta+\tau_2), & \theta\in[-\tau_2,-\tau_1),\end{cases}$$

where $\delta(\theta)$ is the Dirac function defined and Id_3 is the 3×3 identity matrix.

Now, for $\phi\in C([-\tau_2,0],R^3)$, we define

$$(A_\sigma\phi)(\theta)=\begin{cases}\frac{d\phi}{d\theta} & ,\theta\in[-\tau_2,0),\\ \int_{-\tau_2}^{0}d\eta(\xi,\sigma)\phi(\xi)=L_\sigma(\phi) & ,\theta=0,\end{cases}\tag{20}$$

and

$$(R_\sigma\phi)(\theta)=\begin{cases}(0,0,0)^T & ,\theta\in[-\tau_2,0),\\ F(\phi,\sigma) & ,\theta=0.\end{cases}\tag{21}$$

System(16) can be rewritten as

$$\dot{u_t}=A_\sigma u_t+R_\sigma u_t.\tag{22}$$

For $\psi\in C([0,\tau_2],R^3)$, defined

$$(A_0^*\psi)(s)=\begin{cases}-\frac{d\psi}{ds}, & s\in(0,\tau_2],\\ \int_{-\tau_2}^{0}d\eta^T(t,0)\psi(-t), & s=0,\end{cases}\tag{23}$$

and a bilinear inner product

$$\langle\psi,\phi\rangle=\bar{\psi}(0)\phi(0)-\int_{-\tau_2}^{0}\int_{\xi=0}^{\theta}\bar{\psi}^T(\xi-\theta)d\eta(\theta,0)\phi(\xi)d\xi,\tag{24}$$

where $\eta(\theta)=\eta(\theta,0)$. Then A_0 and A_0^* are adjoint operators. By Theorem 3, we know that $\pm i\omega_0$ are the eigenvalues of A_0. Therefore, they are also eigenvalues of A_0^*.

Let $q(\theta)=(1,q_1,q_2)^Te^{i\omega_0\theta}(\theta\in[-\tau_2,0])$ be an eigenvector of A_0 corresponding to the eigenvalue $i\omega_0$, $q^*(s)=D(1,q_1^*,q_2^*)e^{is\omega_0}$ be the eigenvector of A_0^* corresponding to $-i\omega_0$. Denote $V=(1,q_1,q_2)^T, V^*=(1,q_1^*,q_2^*)$. By (24) and $\langle q^*(s),q(\theta)\rangle=1$, we get

$$\bar{D}=[\overline{V^*}^TV+\tau_2e^{-i\omega_0\tau_2}\overline{V^*}^TB_2V+\tau_1e^{-i\omega_0\tau_1}\overline{V^*}^TBV+\sigma e^{-i\omega_0\sigma}\overline{V^*}^TB_1V]^{-1}.$$

In the following, we use the algorithms in Hassard et al.[7] and the computation process similar to that in [3,4], we can obtain the coefficients which will be used in determining the important quantities:

$$
\begin{aligned}
g_{20} &= 2\bar{D}[(1+q_1^2\bar{q}_1^*+q_2^2\bar{q}_2^*)l_1e^{-2i\sigma\omega_0}+(l_3q_1^2+l_5q_2^2)e^{-2i\tau_2\omega_0}+(\bar{q}_1^*m_1+\bar{q}_2^*n_1)e^{-2i\tau_1\omega_0}]\\
g_{11} &= 2\bar{D}(l_1+l_3q_1\bar{q}_1+l_5q_2\bar{q}_2+\bar{q}_1^*m_1+\bar{q}_1^*l_1q_1\bar{q}_1+\bar{q}_2^*n_1+\bar{q}_2^*l_1q_2\bar{q}_2)\\
g_{02} &= 2\bar{D}[(1+\bar{q}_1{}^2\bar{q}_1{}^*+\bar{q}_2{}^2\bar{q}_2{}^*)l_1e^{2i\sigma\omega_0}+(l_3\bar{q}_1{}^2+l_5\bar{q}_2{}^2)e^{2i\tau_2\omega_0}+(\bar{q}_1{}^*m_1+\bar{q}_2{}^*n_1)e^{2i\tau_1\omega_0}]\\
g_{21} &= 2\bar{D}[(2l_1W_{11}^{(1)}(-\sigma)+2l_1\bar{q}_1{}^*q_1W_{11}^{(2)}(-\sigma)+2l_1\bar{q}_2{}^*q_2W_{11}^{(3)}(-\sigma)+l_2+l_2\bar{q}_1{}^*q_1^2\bar{q}_1+\\
&\quad l_2\bar{q}_2{}^*q_2^2\bar{q}_2)e^{-i\sigma\omega_0}+(l_1W_{20}^{(1)}(-\sigma)+l_1\bar{q}_1{}^*\bar{q}_1W_{20}^{(2)}(-\sigma)+l_1\bar{q}_2{}^*\bar{q}_2W_{20}^{(3)}(-\sigma))e^{i\sigma\omega_0}+\\
&\quad (2l_3W_{11}^{(2)}(-\tau_2)q_1+l_4q_1^2\bar{q}_1+2l_5W_{11}^{(3)}(-\tau_2)q_2+l_6q_2^2\bar{q}_2)e^{-i\tau_2\omega_0}+(l_3W_{20}^{(2)}(-\tau_2)\bar{q}_1\\
&\quad +l_5W_{20}^{(3)}(-\tau_2)\bar{q}_2)e^{i\tau_2\omega_0}+(2\bar{q}_1{}^*m_1W_{11}^{(1)}(-\tau_1)+\bar{q}_1{}^*m_2+2\bar{q}_2{}^*n_1W_{11}^{(1)}(-\tau_1)\\
&\quad +\bar{q}_2{}^*n_2)e^{-i\tau_1\omega_0}+(\bar{q}_1{}^*m_1W_{20}^{(1)}(-\tau_1)+\bar{q}_2{}^*n_1W_{20}^{(1)}(-\tau_1))e^{i\tau_1\omega_0}],
\end{aligned}
\tag{25}
$$

where

$$
W_{20}(\theta)=\frac{ig_{20}}{\omega_0}q(0)e^{i\omega_0\theta}+\frac{i\bar{g}_{02}}{3\omega_0}\bar{q}(0)e^{-i\omega_0\theta}+E_1e^{2i\omega_0\theta}, \tag{26}
$$

$$
W_{11}(\theta)=-\frac{ig_{11}}{\omega_0}q(0)e^{i\omega_0\theta}+\frac{i\bar{g}_{11}}{\omega_0}\bar{q}(0)e^{-i\omega_0\theta}+E_2, \tag{27}
$$

and

$$
E_1=\triangle^{-1}(\tau,2i\omega_0)\begin{pmatrix} l_1e^{-2i\sigma\omega_0}+l_3q_1^2e^{-2i\tau_2\omega_0}+l_5q_2^2e^{-2i\tau_2\omega_0}\\ m_1e^{-2i\tau_1\omega_0}+l_1q_1^2e^{-2i\sigma\omega_0}\\ n_1e^{-2i\tau_1\omega_0}+l_1q_2^2e^{-2i\sigma\omega_0}\end{pmatrix},
$$

$$
E_2=\triangle^{-1}(\tau,0)\begin{pmatrix} 2l_1+2l_3q_1\bar{q}_1+2l_5q_2\bar{q}_2\\ 2m_1+2l_1q_1\bar{q}_1\\ 2n_1+2l_1q_2\bar{q}_2\end{pmatrix},
$$

where

$$
(k+\lambda)I-B_1e^{-\lambda\sigma}-B_2e^{-\lambda\tau_2}-Be^{-\lambda\tau_1}\overset{\text{def}}{=}\triangle(\tau,\lambda).
$$

Based on the above analysis, we can see that each g_{ij} in (25) is determined by the parameters in system (3). Thus we can compute the following values:

$$
\begin{aligned}
&c_1(0)=\tfrac{i}{2\omega_0}(g_{20}g_{11}-2|g_{11}|^2-\tfrac{1}{3}|g_{02}|^2)+\tfrac{1}{2}g_{21},\\
&\gamma_2=-\tfrac{Re\{c_1(0)\}}{Re\{\lambda'(\sigma_0)\}},\\
&T_2=-\tfrac{1}{\omega_0}[Im\{c_1(0)\}+\gamma_2Im\{\lambda'(\sigma_0)\}],\\
&B_2=2Re\{c_1(0)\},
\end{aligned}
\tag{28}
$$

which determine the quantities of bifurcating periodic solutions on the center manifold at the critical value σ_0, γ_2 determines the directions of the Hopf bifurcation: if $\gamma_2>0(\gamma_2<0)$, then the Hopf bifurcation is supercritical (subcritical); B_2 determines the stability of the bifurcating periodic solutions: the bifurcating periodic solutions on the center manifold are stable(unstable)if $B_2<0(B_2>0)$; and T_2 determines the period of the bifurcating periodic solutions: the period increase (decrease) if $T_2>0(T_2<0)$.

According to the above analysis and (28), we obtain the following result.

Theorem 4. *Assume (10) holds and* $Re\left[\frac{d\lambda}{d\sigma}\right]_{\sigma=\sigma_0}\neq 0$. *Let* $c_1(0)$ *be given in (28). Then*

(i) the bifurcating periodic solutions exists for $\sigma = \sigma_0$ and if $\gamma_2 > 0 (< 0)$, the bifurcation is supercritical (subcritical);
(ii) the bifurcating periodic solutions are stable (unstable) if $Re\{c_1(0)\} > 0 (< 0)$;
(iii) T_2 determines the period of the bifurcating periodic solutions:the period increase (decrease) if $T_2 > 0 (T_2 < 0)$.

4 Numerical Simulation

In this section, some numerical results of simulating system (3) are presented . We use the formulas obtained in Section 3 to calculate the Hopf bifurcation of system (3) . We suppose $f(x) = f_i(x) = \tanh(x)(i = 1, 2, 3)$ and $k = 1, M = -2, \alpha_{1j} = 1(j = 2, 3), \alpha_{i1} = 1(i = 2, 3)$, then (3) is changed into:

Example 1. the system as follows:

$$\begin{cases} \dot{x}_1(t) = -x_1(t) - 2\tanh(x_1(t-\sigma)) + \tanh(x_2(t-\frac{\pi}{8})) + \tanh(x_3(t-\frac{\pi}{8})), \\ \dot{x}_2(t) = -x_2(t) - 2\tanh(x_2(t-\sigma)) + \tanh(x_1(t-\frac{3\pi}{8})), \\ \dot{x}_3(t) = -x_3(t) - 2\tanh(x_3(t-\sigma)) + \tanh(x_1(t-\frac{3\pi}{8})). \end{cases} \tag{29}$$

By the simple calculation, we know $-2 < -|1 - \sqrt{2}|$. From the calculations of Section 2, we have: $\sigma_0 = 0.5238$, and $\omega_0 = 3.0594$. Meanwhile, we can get

$$Re[(\frac{d\lambda}{d\sigma})_{\sigma=\sigma_0}]^{-1} = 0.03925 > 0.$$

From Theorem 3, we know that the equilibrium of system(29) is asymptotically stable when $\sigma \in [0, 0.5238)$. According to Fig. 2, the numerical simulation can illustrate the fact at $\sigma = 0.5138$. When σ is increased to the critical value 0.5238, the origin will lose its stability and the Hopf Bifurcation occurs, see Fig.1. When σ continues to increase, a stable periodic solution is bifurcated from the zero solution, see Fig.3.

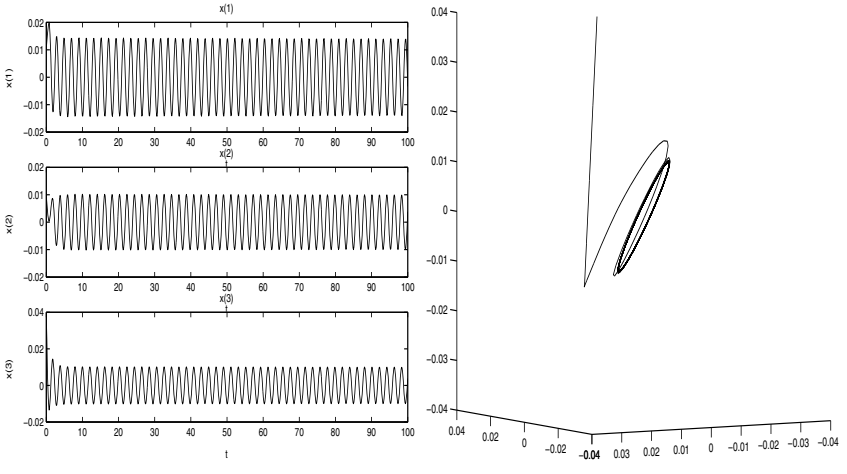


Fig. 1. Numerical solution (x_1, x_2, x_3) of system (29) with $\sigma = 0.5238$

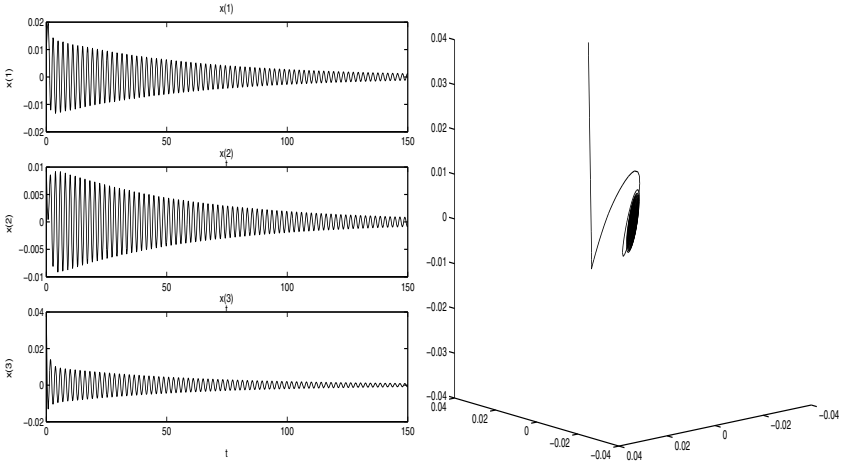


Fig. 2. Numerical solution (x_1, x_2, x_3) of system (29) with $\sigma = 0.5138$

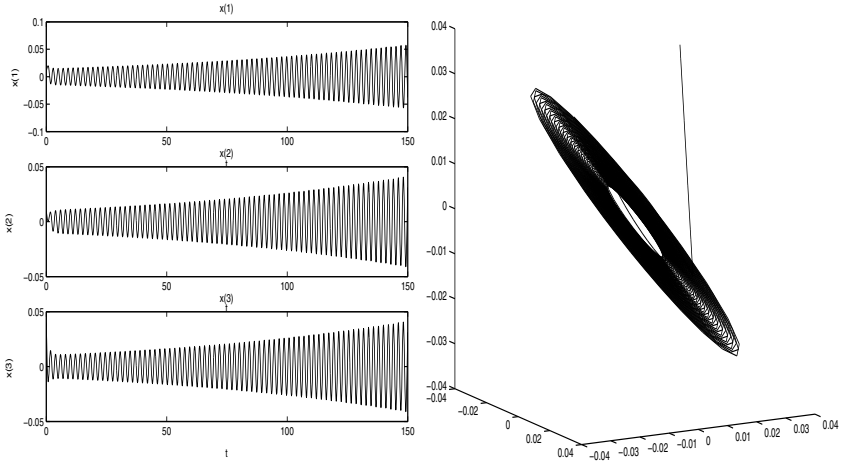


Fig. 3. Numerical solution (x_1, x_2, x_3) of system (29) with $\sigma = 0.5300$

Acknowledgements

The work was supported by National Natural Science Foundation of China under Grants No.60874114 and 60671066.

References

1. Hopfield, J.J.: Neurons with Graded Response Have Collective Computational Properties Like Those of Two-state Neurons. J. Proc. Natl. Acad. Sci. USA 81, 3088–3092 (1984)

2. Marcus, C.M., Westervelt, R.M.: Stability of Analog Neural Network with Delay. J. Phys. Rev. A 39, 347–359 (1989)
3. Wang, L., Zou, X.: Hopf Bifurcation in Bidirectional Associative Memory Neural Networks with Delays: Analysis and Computation. J. Comput. Appl. Math. 167, 73–90 (2004)
4. Song, Y., Han, M., Wei, J.: Stability and Hopf Bifurcation on a Simplified BAM Neural Network with Delays. J. Physics D 200, 185–204 (2005)
5. Driessche, V.D., Wu, P.: Stabilization Role of Inhibitory Self-connections in a Delayed Neural Network. J. Physics D 150, 84–90 (2001)
6. Wang, L.: Stabilizing Hopfield Neural Networks via Inhibitory Self-connections. J. Math. Anal. Appl. 292, 135–147 (2004)
7. Hassard, B.D., Kazarinoff, N.D., Wan, Y.H.: Theory and Applications of Hopf Bifurcation. Cambridge University Press, Cambridge (1981)

Stability Analysis of Recurrent Neural Networks with Distributed Delays Satisfying Lebesgue-Stieljies Measures

Zhanshan Wang, Huaguang Zhang, and Jian Feng

School of Information Science and Engineering, Northeastern University, Shenyang, Liaoning, 110004, People's Republic of China
zhanshan_wang@163.com, wangzhanshan@ise.neu.edu.cn, zhanghuaguang@ise.neu.edu.cn, fengjian@ise.neu.edu.cn

Abstract. Global asymptotic stability problem for a class of recurrent neural networks with a general class of distributed delays has been studied based on distributed-delay-matrix decomposition method and linear matrix inequality (LMI) technique. The proposed stability criterion is suitable for a general class of multiple delayed recurrent neural networks. Especially, for the recurrent neural networks with different multiple delays, infinite distributed delays and finite distributed delays, we have also established corresponding LMI-based stability criteria, which are simple in expression form and easy to check. Compared with the existing results, our results are new and can be regarded as an alterative of M-matrix based stability results in the literature. ...

Keywords: Recurrent neural networks, global asymptotic stability, distributed delays, Lebesgue-Stieljies measures, linear matrix inequality (LMI).

1 Introduction

It is well known that recurrently connected networks have been extensively studied both in theory and applications. They have been successfully applied in signal processing, pattern recognition and associative memories, especially in static image treatment. Such applications heavily rely on the dynamic behaviors of the neural networks. Therefore, the analysis of these dynamic behaviors is a necessary step for the practical design of neural networks [1]-[10].

In hardware implementation, time delays inevitably occur due to the finite switching speed of the amplifiers and communication time. Neural networks with time delays have much more complicated dynamics due to the incorporation of delays. Therefore, stability analysis for recurrent neural networks with different delays have been received many attentions in last decades. In the existing references, different kinds of delays have been considered, for example, discrete or concentrated constant delays τ, τ_j, τ_{ij} and their time varying counterparts; finitely distributed delays $\int_{t-\tau}^{t} g_j(x_j(s))ds$, $\int_{t-\tau_j}^{t} g_j(x_j(s))ds$,

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 504–511, 2010.

$\int_{t-\tau_{ij}}^{t} g_j(x_j(s))ds$ and their time varying counterparts; infinitely distributed delays $\int_{-\infty}^{t} k_{ij}\ (t-s)g_j(x_j(s))ds$, where $g_j(\cdot)$ are the neuron activation functions, $k_{ij}(s)$ are some Kernel functions. Different stability results for recurrent neural networks with above delays have been established, for example, in the form of M-matrix and linear matrix inequality [1]-[10],[23].

Recently, a more general distributed delay model, i.e., $\int_0^{\infty} g_j(x_j(t-s))dK_{ij}(s)$, which is the Lebesgue-Stieljies integral, has been studied in the literature, see [11]-[22]. As pointed in [14,17], can we propose an effective approach to investigate them in a universal framework? An affirmative answer has been given in [14,17] to integrate the different delays based on M-matrix framework. It is well known that linear matrix inequality is a very powerful tool to deal with stability problems associated with different delays. Meanwhile, as a parallel mathematical method to M-matrix in analyzing the stability problem, it is necessary to ask whether LMI based method can also solve the problem proposed in [14,17]. In our previous papers [2,3,4,23], we have established LMI-based stability results for recurrent neural networks with different multiple delays $\tau_{ij}(t)$, infinitely distributed delays $\int_{t-\tau}^{t} g_j(x_j(s))ds$ and infinitely distributed delays $\int_{-\infty}^{t} k_{ij}(t-s)g_j(x_j(s))ds$, respectively. In this paper, we will also give an affirmative answer to the problem proposed in [14,17] and establish an LMI-based stability criterion for recurrent neural neural networks with distributed delays $\int_0^{\infty} g_j(x_j(t-s))dK_{ij}(s)$, which will integrate the cases with above different delays.

2 Problem Description and Preliminaries

The following recurrent neural networks with a general continuously distributed delays will be discussed in this paper,

$$\dot{x}_i(t) = -a_i x_i(t) + \sum_{j=1}^{n} w_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} \int_0^{\infty} f_j(x_j(t-\tau_{ij}(t)-s))dK_{ij}(s), \quad (1)$$

where $x = (x_1, \cdots, x_n)^T$, $A = \mathrm{diag}(a_1, \cdots, a_n)$, $a_i > 0$, $W = (w_{ij})_{n\times n}$ is a real constant matrix, $f(x(t)) = (f_1(x_1(t)), \cdots, f_n(x_n(t)))^T$, $f_i(x_i(t))$ are the activation functions, $\tau_{ij}(t)$ are the time varying delays with $\tau_{ij}(t) \le \tau_M$ and $\dot{\tau}_{ij}(t) \le \mu_{ij} < 1$, $\mu_{ij} > 0$ are positive constants, $dK_{ij}(s)$ are Lebesgue-Stieljies measures for each $i, j = 1, \cdots, n$.

Assumption 1. The activation function $f_i(x_i)$ are bounded and continuous, which satisfy

$$0 \le \frac{f_i(\eta)}{\eta} \le \delta_i, \quad (2)$$

for any $\eta \ne 0, \eta \in \Re$, and $\delta_i > 0$, $f_i(0) = 0$.

Assumption 2. The Lebesgue-Stieljies measures $dK_{ij}(s)$ satisfy

$$\int_0^{\infty} dK_{ij}(s) = m_{ij} > 0, \quad (3)$$

for positive constants $m_{ij} > 0$, $i, j = 1, \cdots, n$.

Inspired by the matrix-decomposition-method proposed in [2,3,4,23], we decompose the distributed delayed terms $\sum_{j=1}^{n}\int_0^\infty f_j(x_j(t-\tau_{ij}(t)-s))dK_{ij}(s)$ in (1) and rewrite (1) in a compact matrix-vector form as follows,

$$\dot{x}(t) = -Ax + Wf(x(t) + \sum_{k=1}^{n} I_k\theta_k(t), \tag{4}$$

where I_k is an $n\times n$ constant matrix, whose elements in k-th row are all 1, and the others rows are all zeros, i.e.,

$$I_1 = \begin{bmatrix} 1 & 1\cdots 1 \\ 0 & 0\cdots 0 \\ \vdots & \vdots\ddots\vdots \\ 0 & 0\cdots 0 \end{bmatrix}_{n\times n}, I_2 = \begin{bmatrix} 0 & 0\cdots 0 \\ 1 & 1\cdots 1 \\ \vdots & \vdots\ddots\vdots \\ 0 & 0\cdots 0 \end{bmatrix}_{n\times n}, I_n = \begin{bmatrix} 0 & 0\cdots 0 \\ 0 & 0\cdots 0 \\ \vdots & \vdots\ddots\vdots \\ 1 & 1\cdots 1 \end{bmatrix}_{n\times n},$$

$$\theta_k(t) = \Big(\int_0^\infty f_1(x_1(t-\tau_{k1}(t)-s))dK_{k1}(s), \int_0^\infty f_2(x_2(t-\tau_{k2}(t)-s))dK_{k2}(s), \cdots, \int_0^\infty f_n(x_1(t-\tau_{kn}(t)-s))dK_{kn}(s)\Big)^T.$$

Lemma 1. (see [23]) For any continuous functions $g_1(s)$ and $g_2(s)$, which are well defined in the integral interval, the following inequality holds,

$$\Big(\int_{-\infty}^{t} g_1(s)g_2(s)ds\Big)^2 \le \Big(\int_{-\infty}^{t} g_1^2(s)ds\Big)\Big(\int_{-\infty}^{t} g_2^2(s)ds\Big), \tag{5}$$

for $t \ge 0$.

3 Main Results

Under above assumptions, we now state our main results in this section.

Theorem 1. Suppose that Assumption 1 and Assumption 2 hold. If there exist positive diagonal matrices P and H_i such that the following linear matrix inequality holds,

$$\Omega = \begin{bmatrix} \Psi & PI_1 & PI_2 & \cdots & PI_n \\ (PI_1)^T & -H_1M_1^{-1}U_1 & 0 & \cdots & 0 \\ (PI_2)^T & 0 & -H_2M_2^{-1}U_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (PI_n)^T & 0 & 0 & \cdots & -H_nM_n^{-1}U_n \end{bmatrix} < 0,$$

then the equilibrium point of neural network (4) is globally asymptotically stable, where $\Psi = -2PA\Delta^{-1} + PW + (PW)^T + \sum_{i=1}^{n} H_iM_i$, $\Delta = \mathrm{diag}(\delta_1, \delta_2, \cdots, \delta_n)$, $M_i = \mathrm{diag}(m_{i1}, m_{i2}, \cdots, m_{in})$ and $U_i = \mathrm{diag}(1-\mu_{i1}, 1-\mu_{i2}, \cdots, 1-\mu_{in})$ are all positive diagonal matrices, $i = 1, \cdots, n$.

Proof. Let us consider the Lyapunov functional $V(t) = V_1(t) + V_2(t)$, where

$$V_1(t) = \sum_{i=1}^{n} 2p_i \int_0^{x_i(t)} f_i(s)ds,$$

$$V_2(t) = \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij} \int_0^{\infty} dK_{ij}(s) \int_{t-\tau_{ij}(t)-s}^{t} f_j^2(x_j(z))dz,$$

where $P = \mathrm{diag}(p_1, \cdots, p_n)$ and $H_i = \mathrm{diag}(h_{i1}, h_{i2}, \cdots, h_{in})$ are all positive diagonal matrices, $i = 1, \cdots, n$.

The derivative of $V_1(t)$ is as follows

$$\begin{aligned}
\dot{V}_1(t) =& \ 2f^T(x(t))P\dot{x}(t) \\
=& \ 2f^T(x(t))P\Big[-Ax(t) + Wf(x(t) + \sum_{k=1}^{n} I_k\theta_k(t)\Big] \\
\le& \ 2f^T(x(t))P\Big[-A\Delta^{-1}f(x(t)) + Wf(x(t) + \sum_{k=1}^{n} I_k\theta_k(t)\Big].
\end{aligned}$$

The derivative of $V_2(t)$ is as follows

$$\begin{aligned}
\dot{V}_2(t) =& \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij} \int_0^{\infty} \Big[f_j^2(x_j(t)) - (1-\dot{\tau}_{ij}(t))f_j^2(x_j(t-\tau_{ij}(t)-s))\Big]dK_{ij}(s) \\
\le& \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}m_{ij}f_j^2(x_j(t)) \\
& - \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{(1-\mu_{ij})h_{ij}}{m_{ij}} \int_0^{\infty} dK_{ij}(s) \int_0^{\infty} f_j^2(x_j(t-\tau_{ij}(t)-s))dK_{ij}(s) \\
\le& \sum_{i=1}^{n} f^T(x(t))H_iM_if(x(t)) \\
& - \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{(1-\mu_{ij})h_{ij}}{m_{ij}} \Big(\int_0^{\infty} f_j(x_j(t-\tau_{ij}(t)-s))dK_{ij}(s)\Big)^2 \\
=& \sum_{i=1}^{n} f^T(x(t))H_iM_if(x(t)) - \sum_{i=1}^{n} \theta_i^T(t)H_iM_i^{-1}U_i\theta_i(t)
\end{aligned}$$

where $\theta_i(t) = \Big(\int_0^{\infty} f_1(x_1(t-\tau_{i1}(t)-s))dK_{i1}(s), \int_0^{\infty} f_2(x_2(t-\tau_{i2}(t)-s))dK_{i2}(s), \cdots, \int_0^{\infty} f_n(x_1(t-\tau_{in}(t)-s))dK_{in}(s)\Big)^T$, $M_i = \mathrm{diag}(m_{i1}, m_{i2}, \cdots, m_{in})$ and $U_i = \mathrm{diag}(1-\mu_{i1}, 1-\mu_{i2}, \cdots, 1-\mu_{in})$ are all positive diagonal matrices, $i = 1, \cdots, n$.

Combining (6) with (6), it yields

$$\dot{V}(t) \le \zeta^T(t)\Omega\zeta(t) < 0,$$

for any $\zeta(t) \neq 0$, where $\zeta^T(t) = \left(f^T(x(t)) \ \theta_1^T(t), \theta_2^T(t), \cdots, \theta_n^T(t)\right)$ and Ω is defined in (6). According to Lyapunov stability theory, the equilibrium point of neural network (4) is globally asymptotically stable.

As pointed out in [14,17], model (4) includes many delayed neural network models. Next, we will discuss the following three cases.

1) In the case $dK_{ij}(s) = \delta(s)w_{ij}^1 ds$, where $\delta(s)$ is the Dirac-delta function, model (4) is reduced to the system with different multiple time varying delays,

$$\dot{x}_i(t) = -a_i x_i(t) + \sum_{j=1}^{n} w_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} w_{ij}^1 f_j(x_j(t - \tau_{ij}(t))).$$

In this case, we have the following result for system (6), which can be derived in a similar manner to the proof of Theorem 1.

Corollary 1. Suppose that Assumption 2 holds. If there exist positive diagonal matrices P and H_i such that the following linear matrix inequality holds,

$$\Omega_1 = \begin{bmatrix} \Psi_1 & PE_1 & PE_2 & \cdots & PE_n \\ (PE_1)^T & -H_1U_1 & 0 & \cdots & 0 \\ (PE_2)^T & 0 & -H_2U_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (PE_n)^T & 0 & 0 & \cdots & -H_nU_n \end{bmatrix} < 0,$$

then the equilibrium point of neural network (6) is globally asymptotically stable, where $\Psi_1 = -2PA\Delta^{-1} + PW + (PW)^T + \sum_{i=1}^{n} H_i$, $\Delta = \mathrm{diag}(\delta_1, \delta_2, \cdots, \delta_n)$, E_i is a $n \times n$ square matrix, whose i-th row is composed of the i-th row of matrix $W_1 = (w_{ij}^1)$ and the other rows are all zeros, and $U_i = \mathrm{diag}(1 - \mu_{i1}, 1 - \mu_{i2}, \cdots, 1 - \mu_{in})$ are all positive diagonal matrices, $i = 1, \cdots, n$.

When Δ is an identity matrix and $\tau_{ij}(t) = \tau_{ij}$ are constant delays, Corollary 1 is reduced to the main result in [2].

2) In the case $dK_{ij}(s) = k_{ij}(s)w_{ij}^1 ds$ and $\tau_{ij}(t) = 0$, model (4) is reduced to the system with infinitely continuously distributed delays,

$$\dot{x}_i(t) = -a_i x_i(t) + \sum_{j=1}^{n} w_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} w_{ij}^1 \int_0^{\infty} k_{ij}(s) f_j(x_j(t-s)) ds.$$

As far as model (6) is concerned, no LMI-based stability result has been published in the literature. If the kernel function $k_{ij}(s)$ also satisfies Assumption 2, i.e.,

$$\int_0^{\infty} k_{ij}(s) ds = m_{ij} > 0,$$

for positive constants $m_{ij} > 0$, $i, j = 1, \cdots, n$, then we have the following result for system (6).

Corollary 2. Suppose that Assumption 1 and condition (6) hold. If there exist positive diagonal matrices P and H_i such that the following linear matrix inequality holds,

$$\Omega_2 = \begin{bmatrix} \Psi_2 & PE_1 & PE_2 & \cdots & PE_n \\ (PE_1)^T & -H_1M_1^{-1}U_1 & 0 & \cdots & 0 \\ (PE_2)^T & 0 & -H_2M_2^{-1}U_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (PE_n)^T & 0 & 0 & \cdots & -H_nM_n^{-1}U_n \end{bmatrix} < 0,$$

then the equilibrium point of neural network (6) is globally asymptotically stable, where $\Psi_2 = -2PA\Delta^{-1} + PW + (PW)^T + \sum_{i=1}^n H_iM_i$, $\Delta = \mathrm{diag}(\delta_1, \delta_2, \cdots, \delta_n)$, E_i is a $n \times n$ square matrix, whose i-th row is composed of the i-th row of matrix $W_1 = (w_{ij}^1)$ and the other rows are all zeros, $M_i = \mathrm{diag}(m_{i1}, m_{i2}, \cdots, m_{in})$ and $U_i = \mathrm{diag}(1 - \mu_{i1}, 1 - \mu_{i2}, \cdots, 1 - \mu_{in})$ are all positive diagonal matrices, $i = 1, \cdots, n$.

3) In the case of 2), if the delay kernel function $k_{ij}(s)$ are of the form $k_{ij}(s) = L_{ij}(s)$ if $s \in [0, r_{ij}]$, otherwise, $k_{ij}(s) = 0$, then the duration intervals for time delays are finite and $\int_0^{r_{ij}} L_{ij}(s)ds = m_{ij} > 0$, $r_{ij} > 0$. Thus, model (4) is reduced to the following neural networks with finite distributed delays,

$$\dot{x}_i(t) = -a_ix_i(t) + \sum_{j=1}^n w_{ij}f_j(x_j(t)) + \sum_{j=1}^n w_{ij}^1 \int_{t-r_{ij}}^t L_{ij}(t-s)f_j(x_j(s))ds.$$

If we further take a special form of the delay kernel function as $L_{ij}(s) = m_{ij}/r_{ij}$, then model (6) can be reduced to the following form

$$\dot{x}_i(t) = -a_ix_i(t) + \sum_{j=1}^n w_{ij}f_j(x_j(t)) + \sum_{j=1}^n w_{ij}^1 m_{ij}/r_{ij} \int_{t-r_{ij}}^t f_j(x_j(s))ds.$$

As far as model (6) is concerned, no LMI-based stability result has been published in the literature. Now we present the following stability result for system (6).

Corollary 3. Suppose that Assumption 1 holds. If there exist positive diagonal matrices P and H_i such that the following linear matrix inequality holds,

$$\Omega_3 = \begin{bmatrix} \Psi_3 & PF_1 & PF_2 & \cdots & PF_n \\ (PF_1)^T & -H_1 & 0 & \cdots & 0 \\ (PF_2)^T & 0 & -H_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (PF_n)^T & 0 & 0 & \cdots & -H_n \end{bmatrix} < 0,$$

then the equilibrium point of neural network (6) is globally asymptotically stable, where $\Psi_3 = -2PA\Delta^{-1} + PW + (PW)^T + \sum_{i=1}^n H_i$, $\Delta = \mathrm{diag}(\delta_1, \delta_2, \cdots, \delta_n)$, F_i is a $n \times n$ square matrix, whose i-th row is composed of the i-th row of matrix $W_{mr} = (w_{ij}^1 m_{ij}/r_{ij})$ and the other rows are all zeros, $i = 1, \cdots, n$.

In summary, the discrete delays $\tau_{ij}(t)$, infinite distributed delays $\int_0^\infty k_{ij}(t-s)$ $f_j(x_j(s))ds$ and finite distributed delays $\int_{t-r_{ij}}^t f_j(x_j(s))ds$ can be included in the model (4) by choosing suitable kernel functions, and above Corollary 1–Corollary 3 can be derived in a similar way to the proof of Theorem 1, respectively.

Now we can revisit the problem proposed in [14,17]. Obviously, parallel to the expression form of stability results based on M-matrix, we have established an alterative form of stability results based on linear matrix inequality. In a unifying framework, we have established a general expression form of stability results for recurrent neural networks with different delays.

4 Conclusions

Inspired by the problem proposed in [14,17] and our previous studies on the recurrent neural networks with different multiple delays [2,3,4,23], we have proposed a novel LMI-based stability result for a class of recurrent neural network with a general class of distributed delays. The proposed stability result has built a unified structure for multiply delayed recurrent neural networks, including multiple discrete delays, infinite distributed delays and finite distributed delays. Therefore, our results will provide a common way to study the nature of different kind of delayed systems.

Acknowledgements. This work was supported by the National Natural Science Foundation of China under the grant No. 50977008, 60774048, and 60774093, by the Specialized Research Fund for the Doctoral Program of Higher Education of China under the grant No. 200801451096, the Postdoctoral Science Foundation of China under the grant No. 20080431150 and 200902547, the Special Funds to Finance Operating Expenses for Basic Scientific Research of Central Colleges of China under the grant No. N090404017, the National High Technology Research and Development Program under the grant 2009AA04Z127, the Program for New Century Excellent Talents in University under the grant No. NCET-08-0101.

References

1. Zeng, Z., Wang, J., Liao, X.: Global Exponential Stability of a General Class of Recurrent Neural Networks with Time-varying Delays. IEEE Trans. Circuits and Systems-I: Fundamental Theory and Applications 50, 1353–1358 (2003)
2. Zhang, H., Wang, Z.: Global Asymptotic Stability of Delayed Cellular Neural Networks. IEEE Trans. Neural Networks 18, 947–950 (2007)
3. Zhang, H., Wang, Z., Liu, D.: Global Asymptotic Stability of Recurrent Neural Networks with Multiple Time Varying Delays. IEEE Trans. Neural Networks 19, 855–873 (2008)
4. Zhang, H., Wang, Z., Liu, D.: Robust Stability Analysis for Interval Cohen-Grossberg Neural Network with Unknown Time-varying Delays. IEEE Trans. Neural Networks 19, 1942–1955 (2008)
5. Cao, J., Yuan, K., Li, H.: Global Asymptotic Stability of Recurrent Neural Networks with Multiple Discrete Delays and Distributed Delays. IEEE Trans. Neural Networks 17, 1646–1651 (2006)

6. Song, Q., Wang, Z.: Neural Networks with Discrete and Distributed Time Varying Delays: a General Stability Analysis. Chaos Solitons and Fractals 37, 1538–1547 (2008)
7. Yu, J., Zhang, K., Fei, S., Li, T.: Simplified Exponential Stability Analysis for Recurrent Neural Networks with Discrete and Distributed Time Varying Delays. Applied Mathematics and Computation 205, 465–474 (2008)
8. Gopalsamy, K., He, X.: Delay-Independent Stability in Bidirectional Asociative Neural Networks. IEEE Trans. Neural Networks 5, 998–1002 (1994)
9. Liang, J., Cao, J.: Global Output Convergence of Recurrent Neural Networkswith Distributed Delays. Nonlinear Analysis: Real World Applications 8, 187–197 (2007)
10. Meng, Y., Guo, S., Huang, L.H.: Convergence Dynamics of Cohen-Grossberg Neural Networks With Continuously Distributed Delays. Applied Mathematics and Computation 202, 188–199 (2008)
11. Chen, T., Lu, W.: Stability Analysis of Dynamical Neural Networks. In: IEEE Int. Conf. Neural Networks and Signal Processing, Nanjing, China, December 14-17 (2003)
12. Lu, W., Chen, T.: On Periodic Dynamical Systems. Chinese Annals of Mathematics Series B 25, 455–462 (2004)
13. Lu, W., Chen, T.: Global Exponential Stability of Almost Periodic Solution for a Large Class of Delayed Dynamical Systems. Science in China, Series A-Mathematics 48, 1015–1026 (2005)
14. Chen, T.: Universal Approach to Study Delay Dynamical Systems. In: Wang, L., Chen, K., S. Ong, Y. (eds.) ICNC 2005. LNCS, vol. 3610, pp. 245–253. Springer, Heidelberg (2005)
15. Chen, T., Lu, W., Chen, G.: Dynamical Behaviors of a Large Class of General Delayed Neural Networks. Neural Computation 17, 949–968 (2005)
16. Chen, T., Lu, W.: Global Asymptotic Stability of Cohen-Grossberg Neural Networks with Time–varying and Distributed Delays. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 192–197. Springer, Heidelberg (2006)
17. Chen, T.: Universal Approach to Study Delayed Dynamical Systems. Studies in Computational Intelligence 35, 85–110 (2007)
18. Lu, W., Chen, T.: Almost Periodic Dynamics of a Class of Delayed Neural Networks with Discontinuous Activations. Neural Computation 20, 1065–1090 (2008)
19. Liao, X., Li, C.: Global Attractivity of Cohen–Grossberg Model with Finite and Infinite delays. Journal of Mathematical Analysis and Applications 315, 244–262 (2006)
20. Liu, P., Yi, F., Guo, Q., Yang, J., Wu, W.: Analysison Global Exponential Robust Stability of Reaction–diffusion Neural Networks With S-Type Distributed Delays. Physica D 237, 475–485 (2008)
21. Wang, L., Xu, D.: Global Asymptotic Stability of Bidirectional Associative Memory Neural Networks With S–Type Distributed Delays. Internat. J. Syst. Sci. 33, 869–877 (2002)
22. Wang, L., Zhang, R., Wang, Y.: Global Exponential Stability of Reaction–Diffusion Cellular Neural Networks with S-Type Distributed Time Delays. Nonlinear Analysis: Real World Applications 10, 1101–1113 (2009)
23. Wang, Z., Zhang, H., Liu, D., Feng, J.: LMI Based Global Asymptotic Stability Criterion for Recurrent Neural Networks with Infinite Distributed Delays. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5551, pp. 463–471. Springer, Heidelberg (2009)

Stability of Genetic Regulatory Networks with Multiple Delays via a New Functional

Zhenwei Liu and Huaguang Zhang

College of Information Science and Engineering, Northeastern University, Shenyang 110819, P.R. China
jzlzw@sina.com, hgzhang@ieee.org

Abstract. This paper studies the stability problem of a class of genetic regulatory networks (GRN) with multiple delays. By using a novel Lyapunov-Krasovskii (L-K) functional based on line integral, the delay-dependent asymptotical stability criterion is first proposed for GRN with multiple delays. The obtained stability result is easy to be checked by linear matrix inequality (LMI) and improve upon the existing ones. Then, a numerical example is given to verify the effectiveness of the proposed criterion.

Keywords: Genetic regulatory networks (GRN), Multiple delays, Asymptotical stability, Linear matrix inequality (LMI), Line integral.

1 Introduction

Genetic regulatory network (GRN) is a collection of DNA segments in a cell. GRN interact with each other indirectly, through their RNA and protein expression products, and with other substances in the cell, thereby governing the rates at which genes in the network are transcribed into mRNA. Each mRNA molecule acts for making a specific protein (or, a set of proteins). In some cases, this protein is structural; while in some other cases, this protein is merely an enzyme that catalyzes a certain chemical reaction. Some proteins only serve to activate the other genes, which are the transcription factors binding to the promoter region at the start of other genes turned on by them, initiating the production of another protein, and so on (see [1]).

Similarly to other dynamic systems (such as recurrent neural networks), stability is a key property for GRN, and some recent results have also been published, see [2]–[6]. It has been recognized that the slow processes of transcription, translation, and translocation or the finite switching speed of amplifiers will inevitable cause time delays, which should be taken into account in the biological systems or artificial genetic networks in order to have more accurate models [7]. It has been shown in [8], by mathematically modelling recent data, that the observed oscillatory expression and activity of three proteins is most likely to be driven by transcriptional delays, and delays can have significant impact both on the dynamical behavior of models and on numerical parameter prediction. Therefore, the effects of transcriptional delays should be assessed in the dynamics of genetic

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 512–519, 2010.

networks whose time scales are short and transcription is regulated by feedback. Especially, when there are the different transcriptional delays in connections among nodes of GRN, the great challenges will be brought. Furthermore, to the best of the authors' knowledge, there is no report on stability analysis of GRN with multiple delays by using Linear Matrix Inequality (LMI).

Therefore, motivated by the above discussions, the new LMI-based asymptotical stability problem for GRN with multiple delays is studied in this paper. By introducing a novel Lyapunov-Krasovskii (L-K) functional based on line integral, it is the first time to propose delay-dependent asymptotically stability criterion for GRN with multiple delays. Meanwhile, a numerical example is given to illustrate the effectiveness of the developed theoretical result.

2 Model and Preliminaries

The following GRN with multiple delays is considered in this paper.

$$\begin{aligned} \dot{m}_i(t) =& - a_i m_i(t) + G_i(p_1(t-\tau_1), p_2(t-\tau_2), \cdots, p_n(t-\tau_n)) \\ \dot{p}_i(t) =& - c_i p_i(t) + d_i m_i(t-\sigma_i) \end{aligned} \tag{1}$$

where τ_i and σ_i denote the constant delays, $i = 1, 2, \cdots, n$. From (1) we can see that, in this network, for any single gene i, there is only one output $p_i(t-\tau_i)$ to other genes, but multiple inputs $p_j(t-\tau_j)$ $(j = 1, 2, \cdots, n)$ from other genes.

As a monotonic increasing or decreasing regulatory function, G_i usually takes the Michaelis-Menten or Hill form. Here, since each transcription factor acts additively to regulate the ith gene or node, the function G_i is taken as $G_i(p_1(t), p_2(t), \cdots, p_n(t)) = \sum_{j=1}^{n} G_{ij}(p_j(t))$, which is called SUM logic [9] [10]. G_i is a monotonic function of the Hill form, that is, if transcription factor j is an activator of gene i, then

$$G_{ij}(p_j(t)) = g_{ij} \frac{\left(\frac{p_j(t)}{\alpha_j}\right)^{H_j}}{1 + \left(\frac{p_j(t)}{\alpha_j}\right)^{H_j}} \tag{2}$$

and if transcription factor j is a repressor of gene i, then

$$G_{ij}(p_j(t)) = g_{ij} \frac{1}{1 + \left(\frac{p_j(t)}{\alpha_j}\right)^{H_j}} = g_{ij} \left(1 - \frac{\left(\frac{p_j(t)}{\alpha_j}\right)^{H_j}}{1 + \left(\frac{p_j(t)}{\alpha_j}\right)^{H_j}} \right) \tag{3}$$

where H_j are the Hill coefficients, α_j is a positive constant, and g_{ij} is a bounded constant, which is the dimensionless transcriptional rate of transcription factor j to i. Based on (2) and (3), the gene networks (3) can be rewritten as

$$\begin{aligned} \dot{m}_i(t) =& - a_i m_i(t) + \sum_{j=1}^{n} b_{ij} \bar{f}_j(p_j(t-\tau_j)) + U_i \\ \dot{p}_i(t) =& - c_i p_i(t) + d_i m_i(t-\sigma_i) \end{aligned} \tag{4}$$

where $\bar{f}_j(p_j(t)) = \left(\frac{p_j(t)}{\alpha_j}\right)^{H_j} / \left(1 + \left(\frac{p_j(t)}{\alpha_j}\right)^{H_j}\right)$, $U_i = \sum_{j\in\mathcal{I}_i} g_{ij}$, and $\mathcal{I}_i$ is the set of all the j which is a repressor of the gene i, $i = 1, 2, \cdots, n$. $B = [b_{ij}] \in \Re^{n\times n}$ is defied as follows:

$$b_{ij} = \begin{cases} g_{ij} & \text{if transcription factor } j \text{ is an activator of gene } i; \\ 0 & \text{if there is no link from gene } j \text{ to gene } i; \\ -g_{ij} & \text{if transcription factor } j \text{ is a repressor of gene } i. \end{cases}$$

The initial condition of the GRN (4) is given by

$$\begin{cases} m(t) = \phi(t) \\ p(t) = \varphi(t) \end{cases} \qquad t \in \left[-\max_{1\leqslant i\leqslant n} (\tau_i, \sigma_i), \quad 0 \right],$$

where $\phi(\cdot)$ and $\varphi(\cdot)$ are continuous functions. Meanwhile, the constant delays σ_i and τ_i satisfy the following condition

$$\begin{cases} \sigma_i > 0, \quad 0 < \sigma_{av} = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} \sigma_i \leqslant \max_{1\leqslant i\leqslant n} (\sigma_i) \\ \tau_i > 0, \quad 0 < \tau_{av} = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} \tau_i \leqslant \max_{1\leqslant i\leqslant n} (\tau_i) \end{cases} \tag{5}$$

For convenience, the model (4) will be written as the following vector-matrix format:

$$\begin{aligned} \dot{m}(t) &= -Am(t) + B\bar{f}(\bar{p}(\bar{t} - \bar{\tau})) + U \\ \dot{p}(t) &= -Cp(t) + D\bar{m}(\bar{t} - \bar{\sigma}) \end{aligned} \tag{6}$$

where $m(t) = [m_1(t), m_2(t), \cdots, m_n(t)]^T$, $p(t) = [p_1(t), p_2(t), \cdots, p_n(t)]^T$, $\bar{t} = tE^T$, $E = [1, 1, \cdots, 1]_{1\times n}$, $\bar{\tau} = [\tau_1, \tau_2, \cdots, \tau_n]^T$ $\bar{p}(\bar{t} - \bar{\tau}) = [p_1(t - \tau_1), p_2(t - \tau_2), \cdots, p_n(t - \tau_n)]^T$ $\bar{f}(\bar{p}(\bar{t} - \bar{\tau})) = [\bar{f}_1(p_1(t - \tau_1)), \bar{f}_2(p_2(t - \tau_2)), \cdots, \bar{f}_n(p_n(t - \tau_n))]^T$, $\bar{\sigma} = [\sigma_1, \sigma_2, \cdots, \sigma_n]^T$, $\bar{m}(\bar{t} - \bar{\sigma}) = [m_1(t - \sigma_1), m_2(t - \sigma_2), \cdots, m_n(t - \sigma_n)]^T$, $A = diag(a_1, a_2, \cdots, a_n)$, $B = [b_{ij}]_{n\times n}$, $C = diag(c_1, c_2, \cdots, c_n)$, $D = diag(d_1, d_2, \cdots, d_n)$, and $U = [U_1, U_2, \cdots, U_n]$.

Remark 1. A vector-matrix model (6) is given by constructing the vectors $\bar{\sigma}$ and $\bar{\tau}$. Namely, the multiple delays σ_i and τ_i can be expressed by vectors $\bar{\sigma}$ and $\bar{\tau}$. Compared with the existing methods, it is more convenient to handle the stability problem for GRN with multiple delays by using LMI technique.

Let $m^\star = [m_1^\star, m_2^\star, \cdots, m_n^\star]^T$ and $p^\star = [p_1^\star, p_2^\star, \cdots, p_n^\star]^T$ be an equilibrium of (6). Thus, we can obtain the following equations

$$\begin{aligned} -Am^\star + B\bar{f}(p^\star) + U = 0, \\ -Cp^\star + Dm^\star = 0. \end{aligned} \tag{7}$$

Define $x_i(\cdot) = m_i(\cdot) - m_i^\star$ and $y_i(\cdot) = p_i(\cdot) - p_i^\star$, system (6) can then be transformed into the following form:

$$\begin{aligned} \dot{x}(t) &= -Ax(t) + Bf(\bar{y}(\bar{t} - \bar{\tau})) \\ \dot{y}(t) &= -Cy(t) + D\bar{x}(\bar{t} - \bar{\sigma}) \end{aligned} \tag{8}$$

where $x(t) = [x_1(t), x_2(t), \cdots, x_n(t)]^T$, $y(t) = [y_1(t), y_2(t), \cdots, y_n(t)]^T$, $f_i(y(t)) = \bar{f}_i(y_i(t) + y_i^\star) - \bar{f}_i(y_i^\star)$, $f(\bar{y}(\bar{t} - \bar{\tau})) = [f_1(y_1(t - \tau_1)), f_2(y_2(t - \tau_2)), \cdots, f_n(y_n(t - \tau_n))]^T$, and $\bar{x}(\bar{t} - \bar{\sigma}) = [x_1(t - \sigma_1), x_2(t - \sigma_2), \cdots, x_n(t - \sigma_n)]^T$.

Since f_i is a monotonically increasing function with saturation, it satisfies the following constraint:

$$0 \leqslant \frac{\bar{f}_i(u) - \bar{f}_i(v)}{u - v} \leqslant l_i \tag{9}$$

for all $u \neq v \in \Re$. Thus, it is obvious that the function $f_i(\cdot)$ satisfies the following condition

$$0 \leqslant \frac{f_i(u)}{u} \leqslant l_i \tag{10}$$

3 Main Results

In this section, the new stability criterion for GRN (8) with multiple delays will be proposed. Since there are two vector delays $\bar{\tau}$ and $\bar{\sigma}$ (i.e., multiple delays) in model (8), a novel L-K functional based on line integral will be used in the stability criterion. Then, the new criterion can be given as follows.

Theorem 1. *The GRN (8) with multiple delays σ_i and τ_i satisfying (5) is asymptotically stable, for given σ_{av} and τ_{av}, if there exist matrices $Z = Z^T = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{12}^T & Z_{22} \end{bmatrix} > 0$, $Q = Q^T > 0$, $W_1 = diag(w_{11}, w_{12}, \cdots, w_{1n}) > 0$, $W_2 = diag(w_{21}, w_{22}, \cdots, w_{2n}) > 0$, $R_1 = R_1^T > 0$, $R_2 = R_2^T > 0$, and scalar parameters $\beta_1 > 0$ and $\beta_2 > 0$, such that the following LMI holds:*

$$\Omega + \sigma_{av}^2 \beta_1 \bar{A}^T \bar{A} + \tau_{av}^2 \beta_2 \bar{C}^T \bar{C} < 0 \tag{11}$$

where

$$\Omega = \begin{bmatrix} \Omega_{11} & Z_{12}D + \frac{\beta_1}{n^3}E^TE & \Omega_{13} & 0 & 0 & Z_{11}B \\ * & -Q - \frac{\beta_1}{n^3}E^TE & DZ_{22} & 0 & 0 & 0 \\ * & * & \Omega_{33} & \frac{\beta_2}{n^3}E^TE & LW_1 & Z_{12}^TB \\ * & * & * & -R_1 - \frac{\beta_2}{n^3}E^TE & 0 & LW_2 \\ * & * & * & * & R_2 - 2W_1 & 0 \\ * & * & * & * & * & -R_2 - 2W_2 \end{bmatrix}$$

$$\Omega_{11} = -Z_{11}A - AZ_{11} + Q - \frac{\beta_1}{n^3}E^TE$$

$$\Omega_{13} = -Z_{12}C - AZ_{12}$$

$$\Omega_{33} = -Z_{22}C - CZ_{22} + R - \frac{\beta_2}{n^3}E^TE$$

$$E = [1, \ 1, \ \cdots, \ 1]_{1\times n}, \quad L = diag(l_i),$$

$$\bar{A} = [-A, \quad 0, \quad 0, \quad 0, \quad 0, \quad B], \quad \bar{C} = [0, \quad D, \quad -C, \quad 0, \quad 0, \quad 0]$$

$i = 1, 2, \cdots, n$, n is the dimension of system (8), and $*$ denotes the symmetric term in a symmetric matrix.

Proof. Consider the following L-K functional:

$$V(t) = V_1(t) + V_2(t) + V_3(t) + V_4(t) + V_5(t), \tag{12}$$

where

$$\begin{aligned}
V_1(t) =& \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}^T Z \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}, \\
V_2(t) =& \frac{1}{n} \int_{C_1} \bar{x}^T(\eta) Q \bar{x}(\eta) E d\eta, \\
V_3(t) =& \frac{1}{n} \int_{C_2} \bar{y}^T(\eta) R_1 \bar{y}(\eta) E d\eta + \frac{1}{n} \int_{C_2} f^T(\bar{y}(\eta)) R_2 f(\bar{y}(\eta)) E d\eta, \\
V_4(t) =& \frac{\sigma_{av}}{n} \beta_1 \int_{C_1} \dot{\bar{x}}^T(\eta) \dot{\bar{x}}(\eta) (\eta - \bar{t} + \bar{\sigma})^T d\eta, \\
V_5(t) =& \frac{\tau_{av}}{n} \beta_2 \int_{C_2} \dot{\bar{y}}^T(\eta) \dot{\bar{y}}(\eta) (\eta - \bar{t} + \bar{\tau})^T d\eta,
\end{aligned}$$

where C_1 and C_2 denote two different integral paths, where C_1 is from $[t - \sigma_1, t - \sigma_2, \cdots, t - \sigma_n]^T$ to $[t, t, \cdots, t]^T$ and C_2 is from $[t - \tau_1, t - \tau_2, \cdots, t - \tau_n]^T$ to $[t, t, \cdots, t]^T$, i.e.,

$$\begin{aligned}
C_1 =& \{[c_{11}, c_{12}, \cdots, c_{1n}]^T | c_{1i} : t - \sigma_i \longrightarrow t, i = 1, 2, \cdots, n\}, \\
C_2 =& \{[c_{21}, c_{22}, \cdots, c_{2n}]^T | c_{2i} : t - \tau_i \longrightarrow t, i = 1, 2, \cdots, n\}.
\end{aligned}$$

Calculating the time derivatives of $V_k(x(t))$ $(k = 1, 2, 3, 4, 5)$ along the trajectories of system (8) yields

$$\dot{V}_1(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}^T Z \begin{bmatrix} -Ax(t) + Bf(\bar{y}(\bar{t} - \bar{\tau})) \\ -Cy(t) + D\bar{x}(\bar{t} - \bar{\sigma}) \end{bmatrix} \tag{13}$$

$$\dot{V}_2(t) = x^T(t) Q x(t) - \bar{x}^T(\bar{t} - \bar{\sigma}) Q \bar{x}(\bar{t} - \bar{\sigma}) \tag{14}$$

$$\begin{aligned}
\dot{V}_3(t) =& y^T(t) R_1 y(t) - \bar{y}^T(\bar{t} - \bar{\tau}) R_1 \bar{y}(\bar{t} - \bar{\tau}) + f^T(y(t)) R_2 f(y(t)) \\
& - f^T(\bar{y}(\bar{t} - \bar{\tau})) R_2 f(\bar{y}(\bar{t} - \bar{\tau}))
\end{aligned} \tag{15}$$

$$\dot{V}_4(t) = \frac{\sigma_{av}}{n} \beta_1 \dot{x}^T(t) \dot{x}(t) E \bar{\sigma} - \frac{\sigma_{av}}{n} \beta_1 \int_{C_1} \dot{\bar{x}}^T(\eta) \dot{\bar{x}}(\eta) E d\eta \tag{16}$$

$$\dot{V}_5(t) = \frac{\tau_{av}}{n} \beta_2 \dot{y}^T(t) \dot{y}(t) E \bar{\tau} - \frac{\tau_{av}}{n} \beta_2 \int_{C_2} \dot{\bar{y}}^T(\eta) \dot{\bar{y}}(\eta) E d\eta \tag{17}$$

Then, by using Jensen's inequality for vector, we have

$$\dot{V}_4(t) \leqslant \sigma_{av}^2 \beta_1 \dot{x}^T(t) \dot{x}(t) - \frac{1}{n^2} \beta_1 \int_{C_1} \dot{\bar{x}}^T(\eta) E d\eta \int_{C_1} \dot{\bar{x}}(\eta) E d\eta \tag{18}$$

$$\dot{V}_5(t) \leqslant \tau_{av}^2 \beta_2 \dot{y}^T(t) \dot{y}(t) - \frac{1}{n^2} \beta_2 \int_{C_2} \dot{\bar{y}}^T(\eta) E d\eta \int_{C_2} \dot{\bar{y}}(\eta) E d\eta \tag{19}$$

Because of $\left(\int_{C_1}\dot{x}_i(\eta_i)Ed\eta\right)^2 \geqslant \left(\int_{C_1}\dot{x}_i(\eta_i)d\eta_i\right)^2$ and $\left(\int_{C_2}\dot{y}_i(\eta_i)Ed\eta\right)^2 \geqslant \left(\int_{C_2}\dot{y}_i(\eta_i)d\eta_i\right)^2$ $(i=1,2,\cdots,n)$, we can know that

$$\begin{aligned}\int_{C_1}\dot{\bar{x}}^T(\eta)Ed\eta\int_{C_1}\dot{\bar{x}}(\eta)Ed\eta \geqslant& \sum_{i=1}^{n}\left(\int_{C_1}\dot{x}_i(\eta_i)d\eta_i\right)^2\\ \geqslant& \frac{1}{n}\int_{C_1}\dot{\bar{x}}^T(\eta)d\eta\int_{C_1}\dot{\bar{x}}^T(\eta)d\eta, \end{aligned}\tag{20}$$

$$\begin{aligned}\int_{C_2}\dot{\bar{y}}^T(\eta)Ed\eta\int_{C_2}\dot{\bar{y}}(\eta)Ed\eta \geqslant& \sum_{i=1}^{n}\left(\int_{C_2}\dot{y}_i(\eta_i)d\eta_i\right)^2\\ \geqslant& \frac{1}{n}\int_{C_2}\dot{\bar{y}}^T(\eta)d\eta\int_{C_2}\dot{\bar{y}}^T(\eta)d\eta. \end{aligned}\tag{21}$$

Meanwhile, according to Generalized Stokes's Theorem [11], we can obtain the following equations,

$$x^T(t)E^T-\bar{x}^T(\bar{t}-\bar{\sigma})E^T=\int_{C_1}\dot{\bar{x}}^T(\eta)d\eta \tag{22}$$

$$y^T(t)E^T-\bar{y}^T(\bar{t}-\bar{\tau})E^T=\int_{C_2}\dot{\bar{y}}^T(\eta)d\eta \tag{23}$$

Thus, $\dot{V}_4(t)$ and $\dot{V}_5(t)$ can be expressed as

$$\begin{aligned}\dot{V}_4(t) \leqslant& \sigma_{av}^2\beta_1\dot{x}^T(t)\dot{x}(t)-\frac{1}{n^3}\beta_1\int_{C_1}\dot{\bar{x}}^T(\eta)d\eta\left[\int_{C_1}\dot{\bar{x}}^T(\eta)d\eta\right]^T\\ =& \sigma_{av}^2\beta_1\dot{x}^T(t)\dot{x}(t)-\frac{1}{n^3}\beta_1(x^T(t)-\bar{x}^T(\bar{t}-\bar{\sigma}))E^TE(x(t)-\bar{x}(\bar{t}-\bar{\sigma})) \end{aligned}\tag{24}$$

$$\begin{aligned}\dot{V}_5(t) \leqslant& \tau_{av}^2\beta_2\dot{y}^T(t)\dot{y}(t)-\frac{1}{n^3}\beta_2\int_{C_2}\dot{\bar{y}}^T(\eta)d\eta\left[\int_{C_2}\dot{\bar{y}}^T(\eta)d\eta\right]^T\\ =& \tau_{av}^2\beta_2\dot{y}^T(t)\dot{y}(t)-\frac{1}{n^3}\beta_2(y^T(t)-\bar{y}^T(\bar{t}-\bar{\tau}))E^TE(y(t)-\bar{y}(\bar{t}-\bar{\tau})) \end{aligned}\tag{25}$$

On the other hand, according to (10), for diagonal matrices $W_1>0$ and $W_2>0$, we can obtain that

$$0 \leqslant f^T(y(t))W_1Ly(t)-f^T(y(t))W_1f(y(t)) \tag{26}$$

$$0 \leqslant f^T(\bar{y}(\bar{t}-\bar{\tau}))W_2L\bar{y}(\bar{t}-\bar{\tau})-f^T(\bar{y}(\bar{t}-\bar{\tau}))W_2f(\bar{y}(\bar{t}-\bar{\tau})) \tag{27}$$

Thus, according to (13)–(15) and (24)–(27), the derivative of $V(t)$ is obtained as follows:

$$\dot{V}(t) \leqslant \zeta^T(t)(\Omega+\sigma_{av}^2\beta_1\bar{A}^T\bar{A}+\tau_{av}^2\beta_2\bar{C}^T\bar{C})\zeta(t) \tag{28}$$

where $\zeta^T(t)=[x^T(t)\ \ \bar{x}^T(\bar{t}-\bar{\sigma})\ \ y^T(t)\ \ \bar{y}^T(\bar{t}-\bar{\tau})\ \ f^T(y(t))\ \ f^T(\bar{y}(\bar{t}-\bar{\tau}))]$.

Obviously, if $\Omega+\sigma_{av}^2\beta_1\bar{A}^T\bar{A}+\tau_{av}^2\beta_2\bar{C}^T\bar{C}<0$, the system (8) is asymptotically stable.

Remark 2. In Theorem 1, a novel L-K functional based on line integral is employed to deal with GRN (8) with multiple delays (i.e., delay vectors $\bar{\sigma}$ and $\bar{\tau}$). Since line integral is an effective tool at vector operation, it is obvious that the functional based on line integral is more suitable to deal with stability of system (8) than the common L-K functional.

Remark 3. Obviously, Theorem 1 is a delay-dependent stability criterion, which depends on the average value of delays σ_i and τ_i, i.e., σ_{av} and τ_{av}. It is achieved by employing the functionals V_4 and V_5, Lemma 1, (20) and (21), and Generalized Stokes's Theorem (i.e., (22) and (23)). Furthermore, it is also first time to present the delay-dependent asymptotical stability criterion for GRN with multiple delays.

4 Numerical Example

In this section, a numerical examples are given to verify the criterion proposed in this paper be effective.

Example 1. Consider GRN (8) with the following parameters [4],

$$A = diag(3,3,3), C = diag(2.5,2.5,2.5), D = diag(0.8,0.8,0.8),$$
$$B = \begin{bmatrix} 0 & 0 & -2.5 \\ -2.5 & 0 & 0 \\ 0 & -2.5 & 0 \end{bmatrix}, \bar{f}_i(p_i(t)) = \frac{p_i^2(t)}{1+p_i^2(t)}, U = [2.5,2.5,2.5]^T, \tag{29}$$

According to these parameters, we can know that the Hill coefficient $H_i = 2$, $\alpha_i = 1$, and $l_i = 0.65$, i.e., $L = diag(0.65, 0.65, 0.65)$. Then, we let $\sigma_{av} = 1.5$ and $\tau_{av} = 0.5$, it means that the delays $\sigma_1 + \sigma_2 + \sigma_3 = 4.5s$ and $\tau_1 + \tau_2 + \tau_3 =$

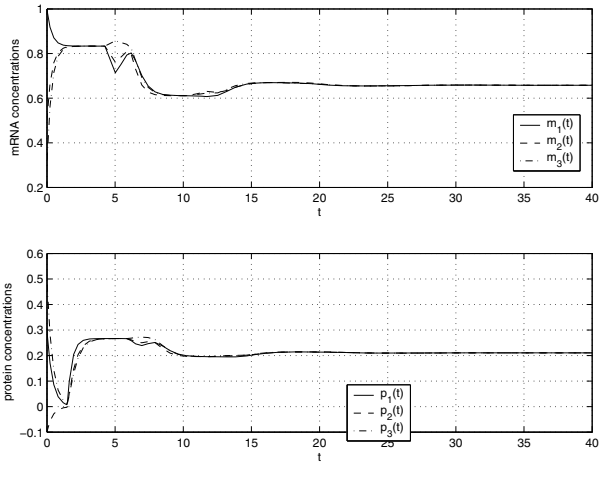


Fig. 1. Transient response of $m_1(t)$, $m_2(t)$, $m_3(t)$, $p_i(t)$, $p_i(t)$, and $p_i(t)$, respectively, when the initial value are $[1, 0.5, 0.3]^T$ and $[0.3, -0.1, 0.5]^T$

1.5s. Following, the stability of GRN with parameters (29) can be verified by using Theorem 1. Fig.1 shows the trajectories of variables $m_1(t)$, $m_2(t)$, $m_3(t)$, $p_i(t)$, $p_i(t)$, and $p_i(t)$, respectively, when the initial value are $[1, 0.5, 0.3]^T$ and $[0.3, -0.1, 0.5]^T$.

5 Conclusion

In this paper, a class of GRN with SUM regulatory logic and multiple delays has been given by exploiting the structure of the genome-proteome network and by representing RNAs and proteins with different variables. For this GRN, a delay-dependent asymptotical stability criterion has been proposed based on LMI technique. It has been carried out by constructing a L-K functional based on line integral. Based on numerical simulation, the obtained criterion can guarantee the asymptotical stability of GRN with multiple delays.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (50977008, 60774048), the Research Fund for the Doctoral Program of China Higher Education (20070145015), the National High Technology Research and Development Program of China (2009AA04Z127).

References

1. Yu, W., Lü, J., Chen, G., Duan, Z., Zhou, Q.: Estimating Uncertain Delayed Genetic Regulatory Networks: An Adaptive Filtering Approach. IEEE Trans. Autom. Contr. 54, 892–897 (2009)
2. Becskel, A., Serrano, L.: Engineering Stability in Gene Networks by Autoregulation. Nature 405, 590–593 (2000)
3. Chen, L., Aihara, K.: Stability of Genetic Regulatory Networks with Time Delay. IEEE Trans. Circu. Syst. I: Fundam. Theory Appl. 49, 602–608 (2002)
4. Ren, F., Cao, J.: Asymptotic and Robust Stability of Genetic Regulatory Networks with Time-Varying Delays. Neurocomputing 71, 834–842 (2008)
5. Cao, J., Ren, F.: Exponential Stability of Discrete-Time Genetic Regulatory Networks with Delays. IEEE Trans. Neural Netw. 19, 520–523 (2008)
6. Wang, Z., Gao, H., Cao, J., Liu, X.: On Delayed Genetic Regulatory Networks With Polytopic Uncertainties: Robust Stability Analysis. IEEE Trans. Nanobioscience 7, 154–163 (2008)
7. Mahaffy, J.M., Pao, C.V.: Models of Genetic Control by Repression with Time Delays and Spatial Effects. J. Math. Biol. 20, 39–57 (1984)
8. Monk, N.A.M.: Oscillatory Expression of Hes1, p53, and NF-κ B Driven by Transcriptional Time Delays. Current Biol. 13, 1409–1413 (2003)
9. Yuh, C.-H., Bolouri, H., Davidson, E.H.: Genomic Cis-Regulatory Logic: Experimental and Computational Analysis of Asea Urchin Gene. Science 279, 1896–1902 (1998)
10. Kalir, S., Mangan, S., Alon, U.: A Coherent Feed-Forward Loop with A SUM Input Function Prolongs Flagella Expression in Escherichia Coli. Molecular Syst. Biol. (2005), doi:10.1038/msb4100010
11. Hubbard, J., Hubbard, B.B.: Vector Calculus, Linear Algebra, and Differential Forms: A Unified Approach, 3rd edn., Matrix, New York, USA (2007)

The Impulsive Control of the Projective Synchronization in the Drive-Response Dynamical Networks with Coupling Delay

Xianyun Xu, Yun Gao, Yanhong Zhao, and Yongqing Yang*

School of Science, Jiangnan University, Wuxi 214122, P.R. China

Abstract. In this paper, we study the impulsive control of the projective synchronization in the drive-response dynamical networks with coupling delay, where the drive system is a partially linear chaotic system and the response system is a delay-coupled dynamical network. The method also allows us to arbitrarily amplify or reduce the scale of the dynamics of the response network through the impulsive control. Numerical simulations are provided to demonstrate the effectiveness of the proposed control method.

Keywords: Projective synchronization, Drive-response dynamical network, Impulsive control, Coupling delay, Chaos.

1 Introduction

Since the pioneering work of Pecora and Carroll [1] in 1990, chaos synchronization has been extensively investigated due to its potential applications in secure communication [2,3,4,5,6]. The typical configuration of chaotic synchronization consists of drive and response systems and has been intensively studied. In 1999, the projective synchronization, a new chaos synchronization phenomena, is first reported by Mainieri and Rehacek [7] in partially linear systems. Under certain conditions, the state vectors of the drive and response systems could be synchronized up to a scaling factor. Scaling factor characterizes the synchronized dynamics of projective synchronization in partially linear chaotic systems but it is difficult to be estimated. To manipulate projective synchronization of chaotic systems in a favored way, in [8,9,10], Xu et al. proposed a control algorithm to direct the scaling factor onto a desired value. Since then, many control methods have been introduced to control the system to the solution of synchronization in a defined way, such as feedback control method [11,12], observer-based approach[13,14], pinning control method[15], impulsive control method [16,17] and so on. Furthermore, the researches on projective synchronization have been extended to general nonlinear systems [18], and time-delayed chaotic systems[19,20], generalized projective synchronization [21,22,23].

* This work was jointly supported by the National Natural Science Foundation of China under Grant 60875036, the Key Research Foundation of Science and Technology of the Ministry of Education of China under Grant 108067 and Supported by Program for Innovative Research Team of Jiangnan University.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 520–527, 2010.

In Ref. [15], Hu et. al. studied projective synchronization in the drive-response dynamical networks. By pinning control techniques, scaling factor can be projection synchronized onto a desired value. Considering the existence of delays due to the finite speed of transmission and spreading as well as congestions, time-delay is a very familiar phenomenon in signal transition. Thereby, time delays should be modelled in order to simulate more realistic networks. Zhao and Yang investigated the impulsive control of projective synchronization in the drive-response complex system[17]. In [28], Cao et. al. discussed the projective synchronization of a class of delayed chaotic systems via impulsive control. To the best of our knowledge, the projective synchronization of the drive-response dynamical networks with coupling delays has not been reported.

In this paper, we study the projective synchronization of a class of the drive-response dynamical networks with coupling delays. The state vectors of the drive and response networks could be synchronized onto a desired scaling factor via the impulsive control, which allows us to arbitrarily amplify or reduce the scale of the dynamics of the response network. Also, using impulsive control method, the respond networks receive the information from the drive system only in discrete times and the amount of conveyed information is, therefore, decreased. This is very advantageous in practice due to reduced control cost.

The organization of this paper is as follows: In Section 2, preliminaries and the drive-response dynamical networks with coupling delays are given. In Section 3, some projective synchronization criteria are obtained based on the stability theory for impulsive functional differential equations. In Section 4, an illustrative example is provided to show the theoretical results. Conclusions are given in Section 5.

2 Model Description and Preliminaries

Consider the following impulsive functional differential equations

$$\begin{cases} \dot{x}(t) = f(t, x_t), & t \neq t_k,\ t \geq t_0, \\ \delta x = x(t_k^+) - x(t_k^-), & t = t_k,\ k = 1, 2, \ldots, \\ x(t_0) = x_0 \end{cases} \tag{1}$$

where $x(t) \in R^n$ is the state variable. $x_t(\theta) = x(t+\theta)$ for $\theta \in [-\tau, 0]$. $f : [t_0, +\infty) \times \Omega \to R^n$ is a smooth nonlinear vector function. $\delta x = x(t_k^+) - x(t_k^-)$ is the control law where $x(t_k^-) = \lim\limits_{t \to t_k^-} x(t)$ and $x(t_k^+) = x(t_k)$ $(k = 1, 2, \ldots)$. The time set $\{t_k\}$ satisfies $0 < t_1 < t_2 < \ldots < t_k < \ldots$, $\lim\limits_{k \to \infty} t_k = \infty$.

We give some lemmas which are needed throughout this paper.

Lemma 1.[26,27] Assume that there exist $V \in v_0$, $w_1, w_2 \in K$, $\varphi \in \kappa$ and $H \in \Gamma$ such that

(1) $w_1(\| x \|) \leq V(t,x) \leq w_2(\| x \|)$ for $(t,x) \in [t_0, +\infty) \times S_\rho$;
(2) for all $x \in S_{\rho_1}, 0 < \rho_1 \leq \rho$, $V(t_k, J_k(x)) \leq \varphi(V(t_k^-, x))$ for all k;
(3) for any solution $x(t)$ of Eq.(1), $V(t+s, x(t+s)) \leq \varphi^{-1}(V(t, x(t)))$, $(-\tau \leq s \leq 0)$, implies that

$$D^+(V(t,x(t))) \leq g(t)H(V(t,x(t))),$$

where $g : [t_0, +\infty) \to R^+$ is locally integrable, φ^{-1} is the inverse function of φ;
(4) H is nondecreasing and there exist constants $l_2 \geq l_1 > 0$ and $r > 0$ such that for any $\mu > 0$, $l_1 \leq t_k - t_{k-1} \leq l_2$ and

$$\int_{\varphi(\mu)}^{\mu} \frac{ds}{H(s)} - \int_{t_{k-1}}^{t_k} g(s)ds \geq r, \ (k = 1, 2, \ldots).$$

Then the zero solution of Eq.(1) is uniformly asymptotically stable.

Lemma 2. If $A = A^T$ is an irreducible real symmetric matrix with $\sum_{j=1}^{N} a_{ij} = 0$, then there exists a unitary matrix $G = (g_{ij})_{N \times N}$ such that $A = G\Xi G^T$, where $G^T G = I$, $\Xi = diag(\lambda_1, \lambda_2, \ldots \lambda_N)$, $0 = \lambda_1 > \lambda_2 \geq \ldots \geq \lambda_N$, λ_i is the eigenvalue of A.

In this paper, we first introduce the drive-response dynamical networks with coupling delays. The network model is described by

$$\begin{cases} \dot{u}_d = M(z) \cdot u_d, \\ \dot{z} = f(u_d, z), \\ \dot{u}_{ri} = M(z) \cdot u_{ri} + c \sum_{j=1}^{N} a_{ij} u_{rj}(t-\tau), \ i = 1, 2, \ldots N, \end{cases} \tag{2}$$

where $\tau \geq 0$ is a bounded constant representing the time delay. $c > 0$ is the coupling strength. $A = (a_{ij})_{N \times N}$ is the coupling configuration matrix that is irreducible and satisfies: $\sum_{j=1}^{N} a_{ij} = 0$, $a_{ij} = a_{ji} \geq 0$ $(i \neq j, \ 1 \leq i \leq N)$. A reflects the topological structures of the whole network. $u_d = (u_d^1, u_d^2 \ldots, u_d^n)^T \in R^n$ is the drive vector, $u_{ri} = (u_{ri}^1, u_{ri}^2, \ldots, u_{ri}^n)^T \in R^n$ is the response state variable. $M(z) = (m_{ij}(z))_{n \times n}$ is a square matrix which depends on the variable z. The equation of z is nonlinear related to the state vector u_d.

If there exists a constant $\alpha (\alpha \neq 0)$ such that $\lim_{t \to \infty} \| u_{ri} - \alpha u_d \| = 0$ for all i, then the projective synchronize of dynamical network (2) is achieved. α is a desired scaling factor. Now, we are interesting in designing an appropriate impulsive controller such that projective synchronize between the drive system and the response networks with coupling delays can achieve, namely, the scaling factor is directed onto a desired value. By introducing the impulse $\delta u_{ri} = d(u_{ri} - \alpha u_d)$ at instant time t_k, the system (2) can be rewritten as follows

$$\begin{cases} \dot{u_d} = M(z) \cdot u_d, \\ \dot{z} = f(u_d, z), \\ \dot{u}_{ri} = M(z) \cdot u_{ri} + c \sum\limits_{j=1}^{N} a_{ij} u_{rj}(t-\tau), & t \neq t_k, \\ \delta u_{ri}(t_k) = d(u_{ri}(t_k^+) - \alpha u_d(t_k^-)), & t = t_k, \end{cases} \tag{3}$$

where d is the control gain.

Define the term $e_i(t) = u_{ri}(t) - \alpha u_d(t)$ as the projective synchronization error of system (3), the error dynamical system can be described as follows

$$\begin{cases} \dot{e}_i = M(z)e_i + c \sum\limits_{j=1}^{N} a_{ij} e_j(t-\tau), & t \neq t_k, \\ \dot{z} = f(u_d, z), \\ e_i(t_k) = (1+d)e_i(t_k^-), & t = t_k,\ i = 1, 2, \ldots N. \end{cases} \tag{4}$$

Then the projective synchronization problem of the dynamical network(2) is equivalent to the problem of stabilization of the error system (4).

Since $A = A^T$ is an irreducible real matrix with $\sum\limits_{j=1}^{N} a_{ij} = 0$, according to Lemma 2, there exists an orthogonal matrix G such that $A = G\Xi G^T$, where $\Xi = diag(\lambda_1, \lambda_2, \ldots \lambda_N)$.

Let $e(t) = (e_1(t), e_2(t), \ldots e_N(t))^T \in R^{N \times n}$, from system (4), one has

$$\dot{e} = eM^T(z) + cAe(t-\tau). \tag{5}$$

Let $e = G\eta$, where $\eta = (\eta_1, \eta_2, \ldots \eta_N)^T$. The error dynamical system (4) is replaced by

$$\begin{cases} \dot{\eta_i} = M(z)\eta_i + c\lambda_i \eta_i(t-\tau), & t \neq t_k, \\ \dot{z} = f(u_d, z), \\ \eta_i(t_k) = (1+d)\eta_i(t_k^-), & t = t_k,\ i = 1, 2, \ldots N. \end{cases} \tag{6}$$

The stability about zero point of linear system (6) implies the stability of origin of the error dynamical system (4).

3 Some Criteria for Projective Synchronization

In this section, some criteria for projective synchronization of the drive-response networks will be established.

Theorem 1. If there exist a set of positive-define matrices $P_i > 0$, a set of positive diagonal matrices $R_i > 0$ and constants $\beta_i > 0$, control gain d such that the following conditions hold:

(**1**) $0 < |1 + d| < 1$,

(**2**) $P_i M(z) + M^T(z)P_i + c\lambda_i^2 P_i R_i P_i + \frac{c\lambda_{max}(R_i^{-1}P_i^{-1})}{(1+d)^2} I \leq \beta_i P_i$,

(**3**) $\max\limits_{1 \leq i \leq N} \{\beta_i\}(t_k - t_{k-1}) + ln(1+d)^2 < 0$.

Then the origin of system (6) is asymptotically stable. That is to say, the projective synchronization of the coupled delayed dynamical drive-response networks (2) is realized, and the scaling factor is the desired value α in advance.

Proof. Considering the following Lyapunov function

$$V(t,\eta)=\sum_{i=1}^{N}\eta_i^T P_i\eta_i.$$

For $\eta\in S(\rho_1),\ 0<\rho_1\le\rho$,

$$V(t_k,(1+d)\eta)=\sum_{i=1}^{N}(1+d)^2\eta_i^T P_i\eta_i=(1+d)^2V(t_k,\eta).$$

Let $\varphi(s)=(1+d)^2 s,\ \varphi\in\kappa$. For any solution of (6), if
$V(t+s,\eta(t+s))\le\varphi^{-1}(V(t,\eta(t))),\ s\in[-\tau,0]$,

specially, for $s=-\tau$, one has
$\sum\limits_{i=1}^{N}\eta_i^T(t-\tau)P_i\eta_i(t-\tau)\le\frac{1}{(1+d)^2}V(t,\eta(t))$.

Calculating the Dini derivative of $V(t,\eta)$ along the trajectories of (6), we have

$$\begin{aligned}D^+V(t,\eta)&=\sum_{i=1}^{N}[M(z)\eta_i+c\lambda_i\eta_i(t-\tau)]^T P_i\eta_i+\sum_{i=1}^{N}\eta_i^T P_i[M(z)\eta_i+c\lambda_i\eta_i(t-\tau)]\\&\le\sum_{i=1}^{N}\eta_i^T[M^T(z)P_i+P_iM(z)]\eta_i+c\sum_{i=1}^{N}[\lambda_i^2\eta_i^T P_iR_iP_i\eta_i+\eta_i^T(t-\tau)R_i^{-1}\eta_i(t-\tau)]\\&\le\sum_{i=1}^{N}\beta_i\eta_i^T P_i\eta_i\le\max_{1\le i\le N}(\beta_i)V(t,\eta(t))\end{aligned}$$

Let $g(t)=1, H(s)=\max\limits_{1\le i\le N}(\beta_i)s$, by the condition (3) in Theorem 1, we have

$$\int_{\varphi(\mu)}^{\mu}\frac{ds}{H(s)}-\int_{t_{k-1}}^{t_k}g(s)ds=\frac{-ln(1+d)^2}{\max\limits_{1\le i\le N}(\beta_i)}-(t_k-t_{k-1})>0.$$

Since P_i is a positive-define matrix, we have

$$\min_{1\le i\le N}(\lambda_{min}(P_i))\sum_{i=1}^{N}\eta_i^T\eta_i\le V(t,\eta)\le\max_{1\le i\le N}(\lambda_{max}(P_i))\sum_{i=1}^{N}\eta_i^T\eta_i.$$

Then the conditions in Lemma 1 are all satisfied. Therefore, the origin of system(6) is asymptotically stable. This completes the proof.

Corollary 1. Let $P_i = R_i = I$ and the impulses be equidistant and separated by interval Δ. Assume that there exist constant β and d such that

(**1**) $0 < |1 + d| < 1$,
(**2**) $M(z) + M^T(z) + c(\lambda_N^2 + \frac{1}{(1+d)^2})I \leq \beta I, \quad \beta > 0$
(**3**) $\beta\Delta + ln(1 + d)^2 < 0$.

Then the origin of the error system (4) is asymptotically stable.

4 Numerical Example on Lorenz System

In this section, to illustrate the effectiveness of results obtained in the previous section, we take a three-dimensional chaotic system as the drive system. Based on the three-dimensional chaotic system, the drive-response networks with coupling delays are described by

$$\begin{cases} \dot{x} = a(y - x), \\ \dot{y} = xz - y, \\ \dot{z} = b - xy - dz, \\ \dot{x_i} = a(y_i - x_i) + c\sum\limits_{j=1}^{5} a_{ij}x_j(t - \tau), \\ \dot{y_i} = x_i z_i - y_i + c\sum\limits_{j=1}^{5} a_{ij}y_j(t - \tau), \quad i = 1, 2, \ldots 5, \end{cases} \tag{7}$$

where the coupling strength $c = 0.2$, time delay $\tau = 0.5$ and the coupling matrix is

$$A = \begin{bmatrix} -3 & 1 & 1 & 0 & 1 \\ 1 & -2 & 0 & 1 & 0 \\ 1 & 0 & -3 & 1 & 1 \\ 0 & 1 & 1 & -3 & 1 \\ 1 & 0 & 1 & 1 & -3 \end{bmatrix},$$

Drive system in (7) is found to be chaotic in a wide parameter range. For example, when $a = 5, \quad d = 1, \quad b = 16$, the drive system is chaotic.

For $M(z) = \begin{pmatrix} -a & a \\ -z & -1 \end{pmatrix}$, after a simple calculation, we have

$$\lambda_{1,2}[M(z) + M^T(z)] = -(1 + a) \pm \sqrt{(1 - a)^2 + (a - z)^2}. \tag{8}$$

Let $m = sup(\lambda[M(z)+M^T(z)])$, by (8), we have $m \leq 7.6015$. Let $d = -1.5, \ m = 7.6015$, the upper bound of the impulsive interval Δ is

$$\Delta < -\frac{2ln|1 + d|}{7.6015 + 0.2(\lambda_5^2 + \frac{1}{(1+d)^2})} = 0.1034.$$

In simulations, the impulsive gain and the impulsive interval are chosen $k = -1.5$ and $\Delta = 0.1$, respectively. Fig.1 shows the simulation results of projective synchronization with the scaling factor $\alpha = 3$. Fig.2 shows the simulation results of anti-synchronization $\alpha = -1$.

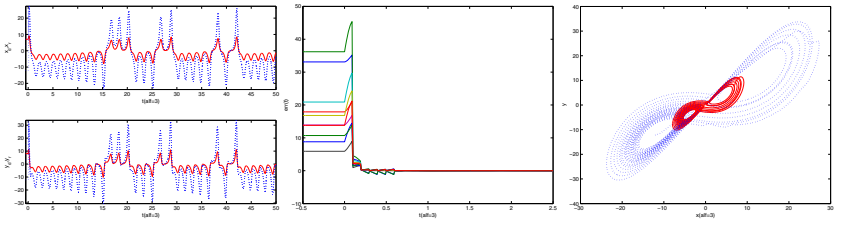


Fig. 1. State trajectories, synchronization errors and phase plot in system (7) with $\alpha = 3$

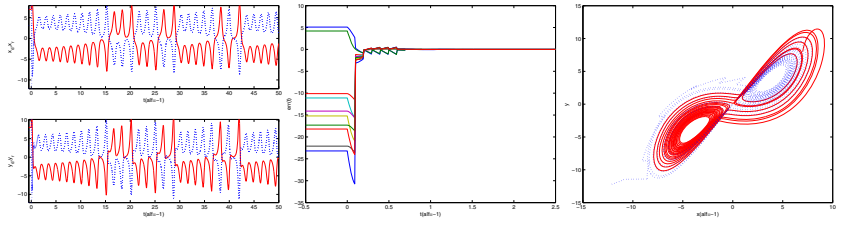


Fig. 2. State trajectories, synchronization errors and phase plot in system (7) with $\alpha = -1$

5 Conclusion

This paper has discussed the projective synchronization in drive-response dynamical networks with coupling delay. Some conditions for the projective synchronization have been derived based on impulsive control method and the stability theory for impulsive functional differential equations. The scaling factor can be arbitrarily amplified or reduced a desired value, that is, the received signals can be changed as demands, which is very a advantageous in applications such as in secure communication. Numerical simulations are given to verify the proposed theoretical results.

References

1. Pecora, L., Carroll, T.: Synchronization in chaotic systems. Phys. Rev. Lett. 64, 821–824 (1990)
2. Cuomo, K.M., Oppenheim, A.V., Strogatz, S.H.: Synchronization of Lorenz-based chaotic circuits with applications to communications. IEEE Trans. Circ. Syst. - II 40, 626–633 (1993)
3. Sundar, S., Minai, A.: Synchronization of randomly multiplexed chaotic systems with application to communication. Phys. Rev. Lett. 85, 5456–5459 (2000)
4. Boccaletti, S., Kurths, J., Osipov, G., Valladares, D.L., Zhou, C.S.: The synchronization of chaotic systems. Phys. Rep. 366, 1–101 (2002)

5. Li, Z., Xu, D.: A secure communication scheme using projective chaos synchronization. Chaos, Soli. Frac. 22, 477–481 (2004)
6. Zhou, J., Chen, T., Xiang, L.: Chaotic lag synchronization of coupled delayed neural networks and its applications in secure communication. Circ. Syst. Sign. Proc. 24, 599–613 (2005)
7. Mainieri, R., Rehacek, J.: Projective synchronization in three-dimensional chaotic systems. Phys. Rev. Lett. 82, 3042–3045 (1999)
8. Xu, D.: Control of projective synchronization in chaotic systems. Phys. Rev. E 63, 027201 (2001)
9. Xu, D., Li, Z.: Manipulating the scaling factor of projective synchronization in three-dimensional chaotic systems. Chaos 11, 439 (2001)
10. Xu, D., Chee, C.: Controlling the ultimate state of projective synchronization in chaotic systems of arbitrary dimension. Phys. Rev. E 66, 046218 (2002)
11. Zou, Y., Zhu, J.: Controlling projective synchronization in coupled chaotic systems. Chinese Phys. 15, 1965–1970 (2006)
12. Hu, M., Xu, Z.: Adaptive feedback controller for projective synchronization. Nonli. Anal.: Real World Appl. 9, 1253–1260 (2008)
13. Wen, G., Xu, D.: Nonlinear observer control for full-state projective synchronization in chaotic continuous-time. Chaos, Soli. Frac. 26, 71–77 (2005)
14. Hung, Y., Hwang, C., Liao, T.: Generalized projective synchronization of chaotic systems with unknown dead-zone input: observer-based approach. Chaos 16, 033125 (2006)
15. Hu, M., Yang, Y., Xu, Z.: Projective synchronization in drive-response dynamical networks. Phys. A 381, 457–466 (2007)
16. Hu, M., Yang, Y., Xu, Z.: Impulsive control of projective synchronization in chaotic systems. Phys. Lett. A 372, 3228–3233 (2008)
17. Zhao, Y., Yang, Y.: The impulsive control synchronization of the drive-response complex system. Phys. Lett. A 372, 7165–7171 (2008)
18. Xu, D., Li, Z.: Controlled projective synchronization in nonparametrically-linear chaotic systems. Int. J. Bifu. Chaos 12, 1395–1402 (2002)
19. Feng, C., Zhang, Y., Wang, Y.: Projective synchronization in time-delayed chaotic systems. Chinese Phys. Lett. 23, 1418–1421 (2006)
20. Feng, C., Zhang, Y., Sun, J., Qi, W., Wang, Y.: Generalized projective synchronization in time-delayed chaotic systems. Chaos, Soli. Frac. 38, 743–747 (2008)
21. Li, G.: Generalized projective synchronization between Lorenz system and Chen's system. Chaos, Soli. Frac. 32, 1454–1458 (2007)
22. Li, G.: Modified projective synchronization of chaotic system. Chaos, Soli. Frac. 32, 1786–1790 (2007)
23. Yan, J., Li, C.: Generalized projective synchronization of a unified chaotic system. Chaos, Soli. Frac. 26, 1119–1124 (2005)
24. Li, C., Chen, G.: Synchronization in general complex dynamical networks with coupling delays. Phys. A 343, 263–278 (2004)
25. Lu, J., Ho, D.: Localand global synchronization in general complex dynamical networks with delay coupling. Chaos, Soli. Fract. 37, 1497–1510 (2008)
26. Li, P., Cao, J.: Robust impulsive synchronization of coupled delayed neural networks with uncertainties. Phys. A 373, 261–272 (2007)
27. Yan, J., Shen, J.: Impulsive stabilization of functional differential equations by Lyapunov-Razumikhin functions. Nonlin. Anal. 37, 245–255 (1999)
28. Cao, J., Ho, D., Yang, Y.: Projective synchronization of a class of delayed chaotic systems via impulsive control. Phys. Lett. A 373, 3128–3133 (2009)

Novel LMI Stability Criteria for Interval Hopfield Neural Networks with Time Delays

Xiaolin Li and Jia Jia

Department of Mathematics, Shanghai University, Shanghai 200444, China

Abstract. In this paper, we consider the uniqueness and global robust stability of the equilibrium point of the interval Hopfield-type delayed neural networks. A new criteria is derived by using linear matrix inequality and Lyapunov functional and also a numerical example is given to show the effectiveness of the present results.

Keywords: Interval Hopfield neural networks, Linear matrix inequality, Lyapunov functional.

1 Introduction

Since J. J. Hopfield proposed the classes of neural networks in [1] and [2], the neural networks with delays (DNN) has attracted much attention due to their application in parallel computation, pattern recognition, associative memories and especially in solving optimization problem.

There are many stability criteria are derived in the present works, for example, asymptotically stability criteria in [3] and [4], and to get more conservative stability criteria, it is of importance to study the robust stability criteria, which are derived in [5]-[12]. Moreover, in this paper, we illustrate a new LMI condition which is more conservative and convenient to verify in practise.

Now we consider a DNN model described by the following functional differential equations:

$$\dot{y_i}(t) = -a_i y_i(t) + \sum_{j=1}^{n} b_{ij} g_j(y_j(t)) + \sum_{j=1}^{n} c_{ij} g_j(y_j(t-\tau)) + u_i, \quad i = 1, 2, \cdots, n. \tag{1}$$

where a_i, $i = 1, 2, \cdots, n$ are positive constants, τ is the transmission delay, b_{ij} and c_{ij}, $i, j = 1, 2, \cdots, n$, denotes the weight coefficients of the neurons, the u_i, $i = 1, 2, \cdots, n$, represent the external constant inputs, and g_j, $j = 1, 2, \cdots, n$, denotes the activation functions. The activation functions are assumed to satisfy the following:

$$0 \leq \frac{g_j(\xi_1) - g_j(\xi_2)}{\xi_1 - \xi_2} \leq L_j, \quad j = 1, 2, \cdots, n. \tag{2}$$

for each $\xi_1, \xi_2 \in R$, $\xi_1 \neq \xi_2$, where $L_j, j = 1, 2, \cdots n$, are positive constants. The quantities a_i, b_{ij}, c_{ij} may be considered as intervalized as following:

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 528–533, 2010.

$$\begin{array}{c} A_I = \{A = diag(a_i) : \underline{A} \le A \le \overline{A}, i.e., \\ \underline{a_i} \le a_i \le \overline{a_i}, i = 1, 2, \cdots, n, \forall A \in A_I\} \\ B_I = \{B = (b_{ij})_{n\times n} : \underline{B} \le B \le \overline{B}, i.e., \\ \underline{b_{ij}} \le b_{ij} \le \overline{b_{ij}}, i, j = 1, 2, \cdots, n, \forall B \in B_I\} \\ C_I = \{C = (c_{ij})_{n\times n} : \underline{C} \le C \le \overline{C}, i.e., \\ \underline{c_{ij}} \le c_{ij} \le \overline{c_{ij}}, i, j = 1, 2, \cdots, n, \forall C \in C_I\}. \end{array} \tag{3}$$

2 Preliminaries

Definition 1 [6]. The system given by (1) with the parameter ranges defined by (3) is globally robust stable if there is a unique equilibrium point $y^* = (y_1^*, y_2^*, \cdots, y_n^*)^T$ of the system, which is globally asymptotically stable for all $A \in A_I$, $B \in B_I$, $C \in C_I$.

Lemma 1 [9]. For any $x = [x_1, x_2, \cdots, x_n]^T$, $y = [y_1, y_2, \cdots, y_n]^T$, $B = (b_{ij})_{n\times n}$, $C = (c_{ij})_{n\times n}$ with $|b_{ij}| \le c_{ij}$, we have

$$x^T By \le |x|^T C|y|. \tag{4}$$

For the proof of the uniqueness and global robust stability of the equilibrium point of system (1), we make the following transformation: Let $y^* = (y_1^*, y_2^*, \cdots, y_n^*) \in R^n$ is the origin of the system, we will always shift an intended equilibrium point y^* of system (1) to the origin by letting $x(t) = y(t) - y^*$, which transform the system (1) into the following system:

$$\dot{x}(t) = -Ax(t) + Bf(x(t)) + Cf(x(t-\tau)). \tag{5}$$

where $x(t) = [x_1(t), x_2(t), \cdots, x_n(t)]^T$ is the state vector of the transformed system, $f(x) = [f_1(x), f_2(x), \cdots, f_n(x)]^T$ denotes the activation function vector $f(x)$ possesses the following properties:

$$0 \le \frac{f_j(x_j)}{x_j} \le L_j, f_j(0) = 0, j = 1, 2 \cdots, n. \tag{6}$$

Clearly, the equilibrium point x^* of (5) with (6) is globally robust stable if and only if the equilibrium point of system (1) with (2) and (3) is globally robust stable. Thus, we only consider the global robust stability of the equilibrium point for system (5).

In the following, for real symmetric matrix X and Y, the notation $X > Y$ (respectively, $X \ge Y$) means that the matrix $X - Y$ is positive definite(respectively, semi-define). For $x = [x_1, x_2, \cdots, x_n]^T \in R^n$, let $|x|$ denote the absolute-value vector given by $|x| = [|x_1|, |x_2|, \cdots, |x_n|]^T$. For a matrix $A = (a_{ij})_{(n\times n)} \in R^{n\times n}$, let $|A|$ denote the absolute-value matrix given by $|A| = (|a_{ij}|)_{n\times n}$, $L = diag(L_1, L_2, \cdots, L_n)$.

3 Main Result

Theorem 1. Under the assumptions (3), system (5) is globally robust stable if there exits positive diagonal matrices P, Q, Q_1, Q_3, Q_4 and a matrix Q_2, such that the following LMIs holds:

$$\Pi_1 = \begin{bmatrix} -2P\underline{A} + Q + 2Q_1 + L^T Q_4 L & 0 & PB^* + 2Q_2 & PC^* \\ 0 & -Q_1 & 0 & -2Q_2 \\ {B^*}^T P + 2Q_2^T & 0 & -Q_4 + Q_3 & 0 \\ {C^*}^T P & -2Q_2^T & 0 & -Q_3 \end{bmatrix} < 0, \quad (7)$$

and

$$\Pi_2 = \begin{bmatrix} -Q + L^T Q_4 L & 0 & PB_* & PC_* \\ 0 & -Q_1 & 0 & 0 \\ {B_*}^T P & 0 & -Q_4 + Q_3 & 0 \\ {C_*}^T P & 0 & 0 & -Q_3 \end{bmatrix} < 0, \quad (8)$$

and

$$\begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix} > 0, \quad (9)$$

where $B^* = \frac{1}{2}(\overline{B} + \underline{B})$, $B_* = \frac{1}{2}(\overline{B} - \underline{B})$, $C^* = \frac{1}{2}(\overline{C} + \underline{C})$, $C_* = \frac{1}{2}(\overline{C} - \underline{C})$.

Proof. Consider the following positive definite Lyapunov functional:

$$V(x(t)) = x^T(t)Px(t) + 2\int_{t-\tau}^{t} \begin{bmatrix} x(s) \\ f(x(s)) \end{bmatrix}^T \begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix} \begin{bmatrix} x(s) \\ f(x(s)) \end{bmatrix} ds. \quad (10)$$

The time derivative of along the trajectories of (5) takes the form:

$$\begin{aligned} \dot{V}(x(t)) &= 2x^T(t)P\dot{x}(t) + 2\begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix}^T \begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix} \begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix} \\ &\quad -2\begin{bmatrix} x(t-\tau) \\ f(x(t-\tau)) \end{bmatrix}^T \begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ f(x(t-\tau)) \end{bmatrix} \\ &= 2x^T(t)P(-Ax(t) + Bf(x(t)) \\ &\quad +Cf(x(t-\tau))) + 2x^T(t)Q_1x(t) \\ &\quad +2f^T(x(t))Q_2^T x(t) + 2x^T(t)Q_2 f(x(t)) \\ &\quad +2f^T(x(t))Q_3 f(x(t)) \\ &\quad -2x^T(t-\tau)Q_1x(t-\tau) \\ &\quad -2f^T(x(t-\tau))Q_2^T x(t-\tau) \\ &\quad -2x^T(t-\tau)Q_2 f(x(t-\tau)) \\ &\quad -2f^T(x(t-\tau))Q_3 f(x(t-\tau)). \end{aligned}$$

Since a_i, p_i, $i = 1, 2, \cdots, n$, are positive constants, we can get that

$$-2x_i(t)p_i a_i x_i(t) \leq -2x_i(t)p_i \underline{a_i} x_i(t), i = 1, 2 \cdots, n. \tag{11}$$

i.e.

$$-2x^T(t)PAx(t) \leq -2x^T(t)P\underline{A}x(t). \tag{12}$$

Also, there is a matrix Q_4, such that

$$2f^T(x(t))Q_4 f(x(t)) \leq 2x^T(t)L^T Q_4 Lx(t). \tag{13}$$

Then we rewrite $x^T(t)PBx(t)$ as

$$x^T(t)PBf(x(t)) = x^T(t)PB^* f(x(t)) + x^T(t)PB_0 f(x(t)). \tag{14}$$

where $B_0 = B - B^*$. From (3), we can get $|b_{0ij}| \leq b_{*ij}$. since $p_i \geq 0, i = 1, 2, \cdots, n$, we can get $|p_i b_{0ij}| \leq p_i b_{*ij}$. It can get from Lemma 1 that

$$x^T(t)PB_0 f(x(t)) \leq |x(t)|^T PB_* |f(x(t))|. \tag{15}$$

So, we have

$$x^T(t)PBf(x(t)) \leq x^T(t)PB^* f(x(t)) + |x(t)|^T PB_* |f(x(t))|. \tag{16}$$

Similarly, we have

$$x^T(t)PCf(x(t-\tau)) \leq x^T(t)PC^* f(x(t-\tau)) + |x(t)|^T PC_* |f(x(t-\tau))|. \tag{17}$$

Obviously, it is easy to see that

$$\begin{gathered} x^T(t)Qx(t) = |x(t)|^T Q|x(t)|, \\ f^T(x(t))Q_3 f(x(t)) = |f(x(t))|^T Q_3 |f(x(t))|, \\ f^T(x(t-\tau))Q_3 f(x(t-\tau) = |f(x(t-\tau))|^T Q_3 |f(x(t-\tau))|. \end{gathered} \tag{18}$$

Then, by using (11) – (18), we can get that

$$\begin{aligned} \dot{V}(x(t)) \leq\ & x^T(t)(-2P\underline{A} + Q)x(t) \\ & +2x^T(t)PB^* f(x(t)) + 2|x(t)|^T PB_* \\ & \times |f(x(t))| + 2x^T(t)PC^* f(x(t-\tau)) \\ & +2|x(t)|^T PC_* |f(x(t-\tau))| \\ & +2x^T(t)Q_1 x(t) + 2f^T(x(t))Q_2^T x(t) \\ & +2x^T(t)Q_2 f(x(t)) + 2f^T(x(t))Q_3 f(x(t)) \\ & -2x^T(t-\tau)Q_1 x(t-\tau) \\ & -2f^T(x(t-\tau))Q_2^T x(t-\tau) \\ & -2x^T(t-\tau)Q_2 f(x(t-\tau)) \\ & -2f^T(x(t-\tau))Q_3 f(x(t-\tau)) \\ & -x^T(t)Qx(t) + 2x^T(t)L^T Q_4 Lx(t) \\ & -2f^T(x(t))Q_4 f(x(t)) \\ =\ & \Theta^T(t)\Pi_1 \Theta(t) + |\Theta(t)|^T \Pi_2 |\Theta(t)|. \end{aligned}$$

where $\Theta(t) = [x^T(t), x^T(t-\tau), f^T(x), f^T(x(t-\tau))]^T$. Then from the LMIs, we can get that $\dot{V}(x(t)) < 0$. So the origin of (5) is globally robust stable.

Now we will prove the uniqueness of the equilibrium. The equilibrium point x is given by

$$-Ax + Bf(x) + Cf(x) = 0. \tag{19}$$

Assume that there exists an nonsingular equilibrium point satisfying (19), we define

$$\begin{aligned} F(x) &= 2x^T P[-Ax + Bf(x) + Cf(x)] + x^T(t)Qx(t) - x^T Qx(t) \\ &\quad +2\begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix}^T \begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix} \begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix} \\ &\quad -2\begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix}^T \begin{bmatrix} Q_1 & Q_2 \\ Q_2^T & Q_3 \end{bmatrix} \begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix}; \\ G(x) &= 2x^T(t)L^T Q_4 Lx(t) - 2f^T(x(t))Q_4 f(x(t)). \end{aligned} \tag{20}$$

We can see that $F(x) = 0$ and $G(x) \geq 0$, so $F(x) + G(x) \geq 0$. On the other hand,

$$F(x) + G(x) \leq \overline{\Theta}^T(t)\Pi_1\overline{\Theta}(t) + |\overline{\Theta}(t)|^T \Pi_2 |\overline{\Theta}(t)|. \tag{21}$$

where $\overline{\Theta}(t) = [x^T(t), x^T(t), f^T(x(t)), f^T(x(t))]^T$. From (7) and (8) we can know that $\overline{\Theta}^T(t)\Pi_1\overline{\Theta}(t) + |\overline{\Theta}(t)|^T \Pi_2 |\overline{\Theta}(t)| < 0$, which implies system (5) has a unique equilibrium point x^*.

This proof is complete.

Remark 1: In this paper, a new inequality technique, i.e. Lemma 1 is used. By Lemma 1, we obtain a new sufficient criteria in form of LMI for the uniqueness and global robust stability of the equilibrium point of the interval Cohen-Grossberg neural network with time-varying delays. It is well known that the inequality $\| A \|_2 \leq \| A^* \|_2 + \| A_* \|_2$ is extensively adopted to investigated the robust stability of the neural networks in the existing works, as a result, the obtained criteria contain the terms of $\| A^* \|_2$ and $\| A_* \|_2$. However, in this paper, by virtue of Lemma 1, the obtained results don't contain $\| A^* \|_2$ and $\| A_* \|_2$, therefore, our results are less restrictive than those existing ones.

4 Example

In the following, we will give an example to consider a delayed neural networks:

$$\underline{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \overline{A} = \begin{bmatrix} 1.02 & 0 \\ 0 & 1.01 \end{bmatrix}, \underline{B} = \begin{bmatrix} 0.01 & -0.035 \\ 0.03 & 0.01 \end{bmatrix}, \overline{B} = \begin{bmatrix} 0.03 & -0.025 \\ 0.03 & 0.03 \end{bmatrix},$$
$$\underline{C} = \begin{bmatrix} -0.025 & -0.015 \\ -0.016 & -0.025 \end{bmatrix}, \overline{C} = \begin{bmatrix} 0.075 & 0.085 \\ -0.014 & 0.075 \end{bmatrix}$$

and with $f_j(x_j) = tanh(x_j)$, $j = 1, 2$. So, we can get that:

$$B^* = \begin{bmatrix} 0.02 & -0.03 \\ 0.06 & 0.02 \end{bmatrix}, B_* = \begin{bmatrix} 0.01 & 0.005 \\ 0 & 0.01 \end{bmatrix}, C^* = \begin{bmatrix} 0.025 & 0.035 \\ -0.015 & 0.025 \end{bmatrix},$$
$$C_* = \begin{bmatrix} 0.05 & 0.05 \\ 0.001 & 0.05 \end{bmatrix}, L = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

By using the Matlab LMI Control toolbox, it can be easily verified that the LMIs (7) − (9) is feasible and

$$P = \begin{bmatrix} 27.3638 & 0 \\ 0 & 27.4136 \end{bmatrix}, Q = \begin{bmatrix} 16.2519 & 0 \\ 0 & 16.2750 \end{bmatrix}, Q_1 = \begin{bmatrix} 9.6075 & 0 \\ 0 & 9.6226 \end{bmatrix},$$
$$Q_2 = \begin{bmatrix} -0.1216 & 0.1824 \\ -0.3655 & -0.1218 \end{bmatrix}, Q_3 = \begin{bmatrix} 8.8439 & 0 \\ 0 & 8.8492 \end{bmatrix}, Q_4 = \begin{bmatrix} 13.2887 & 0 \\ 0 & 13.3047 \end{bmatrix}.$$

Acknowledgments. This work was supported by the National Natural Science Foundation of China under Grant No. 10902065.

References

1. Hopfield, J.J.: Neural Networks and Physical Systems with Emergent Collective Computational Abilities. Proc. Nat. Acad. Sci. USA 79, 2554–2558 (1982)
2. Hopfield, J.J.: Neural Networks with Graded Response Have Collective Computational Properties Like Those of Two-stage Neurons. Proc. Nat. Acad. Sci. USA 81, 3088–3092 (1984)
3. Guo, S., Huang, L.: Stability Analysis of Cohen-Grossberg Neural Networks. IEEE Tran. Neural Netw. 17, 106–117 (2006)
4. Chen, T., Rong, L.: Delay-independent Stability Analysis of Cohen-Grossberg Neural Networks. Phys. Lett. A 317, 436–449 (2003)
5. Liao, X., Yu, J.: Robust Stability for Interval Hopfield Neural Networks with Time Delay. IEEE Trans. Neural Networks 9, 1042–1046 (1998)
6. Cao, J., Huang, D., Qu, Y.: Global Robust Stability of Delayed Recurrent Neural Networks. Chaos, Solutions and Fractals 23, 221–229 (2005)
7. Liao, X., Wang, J., Cao, J.: Global and Robust Stability of Interval Hopfield Neural Networks with Time-varying Delays. Int. J. Neural. Syst 13, 171–182 (2004)
8. Liao, X., Wang, K., Wu, Z., Chen, G.: Novel Robust Stability Criteria for Interval-delayed Hopfield Neural Networks. IEEE Trans. Circuits Syst. I 48, 1355–1359 (2001)
9. Shen, T., Zhang, Y.: Improved Global Robust Stabilty Criteria for Delayed Neural Networks. IEEE Trans. Circuits Syst. II, Exp. Brief 54, 715–719 (2007)
10. Arik, S.: Global Robust Stability of Delayed Neural Networks. IEEE Trans. Cricuits Syst. I, fundam. Theory App. 50, 156–160 (2003)
11. Cao, J., Wang, J.: Global Asymptotic and Robust Stability of Recurrent Neural Networks with Time Delay. IEEE Trans. I 52, 417–426 (2005)
12. Ozcan, N., Arik, S.: Global Robust Stability Analysis of Neural Networks with Multiple Time Delays. IEEE Trans. I 53, 166–176 (2005)

Memetic Evolutionary Learning for Local Unit Networks

Roman Neruda and Petra Vidnerová

Institute of Computer Science
Academy of Sciences of the Czech Republic
Pod vodárenskou věží 2, Prague 8, Czech Republic
roman@cs.cas.cz

Abstract. In this work we propose two hybrid algorithms combining evolutionary search with optimization algorithms. One algorithm memetically combines global evolution with gradient descent local search, while the other is a two-step procedure combining linear optimization with evolutionary search. It is shown that these algorithms typically produce smaller local unit networks with performance similar to theoretically sound but large regularization networks.

1 Introduction

The problem of *supervised learning* is extensively studied both in theory and applications. Consider the situation when we are given a set of examples $\{(\boldsymbol{x}_i, y_i) \in R^d \times R\}_{i=1}^N$ obtained by random sampling of some real function f, generally in presence of noise. To this set we refer as *a training set*. The goal is to recover the function f from data, or find the best estimate of it. It is not necessary that the function exactly interpolates all the given data points, but we need a function with good generalization. That is a function that gives relevant outputs also for the data not included in the training set.

The supervised learning is often studied as a function approximation problem [1]. Given the data set, we are looking for the function that approximates the unknown function f. It is usually done by *empirical risk minimization*, i.e. minimizing the functional $H[f] = \frac{1}{N}\sum_{i=1}^N (f(\boldsymbol{x}_i) - y_i)^2$ over a chosen *hypothesis space*, i.e. over a set of functions of a chosen type (representable by a chosen type of neural network).

We studied the relation between the network size and approximation accuracy and generalization by experimental means [2]. With respect to theoretical results, we expect the approximation accuracy to improve with increasing number of hidden units. Reasonable approximation accuracy should be achieved already with small networks. In addition, high number of hidden units makes the learning task more difficult, which can influence the results. Based on these recommendations, we introduce in Section 3 two hybrid learning algorithms combining global search by genetic algorithm and a local search by a gradient descent method or linear optimization, respectively. The performance of the algorithms is demonstrated on experiments in Section 4.

2 Approximation via Regularization Network

We are given a set of examples $\{(\boldsymbol{x}_i, y_i) \in R^d \times R\}_{i=1}^N$ obtained by random sampling of some real function f and we would like to find this function. Since this problem

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 534–541, 2010.

is ill-posed, we have to add some a priori knowledge about the function. We usually assume that the function is *smooth*, in the sense that two similar inputs corresponds to two similar outputs and the function does not oscillate too much. This is the main idea of the regularization theory, where the solution is found by minimizing the functional (1) containing both the data and smoothness information.

$$H[f] = \frac{1}{N}\sum_{i=1}^{N}(f(\boldsymbol{x}_i) - y_i)^2 + \gamma\Phi[f], \tag{1}$$

where Φ is called a *stabilizer* and $\gamma > 0$ is *the regularization parameter* controlling the trade-off between the closeness to data and the smoothness of the solution. The regularization scheme (1) was first introduced by Tikhonov [3] and therefore it is called a Tikhonov regularization. The regularization approach has good theoretical background, it was shown that for a wide class of stabilizers the solution has a form of feed-forward neural network with one hidden layer, called *regularization network*, and that different types of stabilizers lead to different types of regularization networks [4,5].

Poggio and Smale in [5] proposed a learning algorithm (Alg. 2.1) derived from the regularization scheme (1). They choose the hypothesis space as a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_K$ defined by an explicitly chosen, symmetric, positive-definite kernel function $K_{\boldsymbol{x}}(\boldsymbol{x}') = K(\boldsymbol{x}, \boldsymbol{x}')$. The stabilizer is defined by means of norm in $\mathcal{H}_K$, so the problem is formulated as follows:

$$\min_{f\in\mathcal{H}_K} H[f], \text{ where } H[f] = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\boldsymbol{x}_i))^2 + \gamma||f||_K^2. \tag{2}$$

The solution of minimization (2) is unique and has the form

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} w_i K_{\boldsymbol{x}_i}(\boldsymbol{x}), \qquad (N\gamma I + K)\boldsymbol{w} = \boldsymbol{y}, \tag{3}$$

where I is the identity matrix, K is the matrix $K_{i,j} = K(\boldsymbol{x_i}, \boldsymbol{x_j})$, and $\boldsymbol{y} = (y_1, \ldots, y_N)$.

The solution (3) can be represented by a neural network with one hidden layer and output linear layer. The most commonly used kernel function is Gaussian $K(\boldsymbol{x}, \boldsymbol{x}') = e^{-\left(\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|}{b}\right)^2}$.

The power of the above algorithm is in its simplicity and effectiveness. However, its real performance depends significantly on the choice of parameter γ and kernel function type. Optimal choice of these parameters depends on a particular data set and there is no general heuristics for setting them.

An RBF network is a standard feed-forward neural network with one hidden layer of RBF units and linear output layer (Fig. 2). The RBF units represent RBF functions (5), usually Gaussians. The network computes its output (6) as linear combination of outputs of the hidden layer.

There is a variety of algorithms for RBF network learning, in our past work we studied their behavior and possibilities of their combinations [6]. The most common used *Gradient learning* algorithm is sketched bellow, cf. [6] for details.

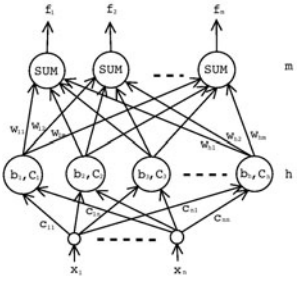
Input: Data set $\{\boldsymbol{x}_i, y_i\}_{i=1}^N \subseteq X \times Y$
Output: Function f.

1. Choose a symmetric, positive-definite function $K_{\boldsymbol{x}}(\boldsymbol{x}')$, continuous on $X \times X$.
2. Create $f : X \to Y$ as $f(\boldsymbol{x}) = \sum_{i=1}^N c_i K_{\boldsymbol{x}_i}(\boldsymbol{x})$ and compute $\boldsymbol{w} = (w_1, \ldots, w_N)$ by solving

$$(N\gamma I + K)\boldsymbol{w} = \boldsymbol{y}, \tag{4}$$

where I is the identity matrix, $K_{i,j} = K(\boldsymbol{x_i}, \boldsymbol{x_j})$, and $\boldsymbol{y} = (y_1, \ldots, y_N), \gamma > 0$.

Fig. 1. Algorithm of regularization network learning



$$y(\boldsymbol{x}) = \varphi\left(\frac{\| \boldsymbol{x} - \boldsymbol{c} \|}{b}\right) \tag{5}$$

$$f_s(\boldsymbol{x}) = \sum_{j=1}^{h} w_{js}\varphi\left(\frac{\| \boldsymbol{x} - \boldsymbol{c}_j \|}{b_j}\right) \tag{6}$$

Fig. 2. a) RBF network architecture b) RBF network function

The solution derived in the previous section contains as many hidden units as is the number of data samples. Such solution is unfeasible for most real life problems. Therefore solutions with lower number of hidden units are considered.

We studied the relation between the network size (i.e. number of hidden units) and approximation accuracy and generalization by experimental means [2].

Since the convergence is quite fast, we can suggest that small networks provide sufficiently good solutions. The theoretically estimated convergence rates justify using network of smaller complexity in real-life applications. Smaller networks have also smaller number of parameters that has to be tuned during the training process. Therefore, they are more easily trained.

3 Two Hybrid Learning Algorithms

In this section we propose a memetic learning algorithm and a two-step hybrid algorithm creating regularization networks with small number of hidden units. It is based on a combination of evolutionary algorithms and gradient descent method, or linear optimization respectively.

Input: Data set $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$
Output: $\{\boldsymbol{c}_i, b_i, C_i, w_{ij}\}_{i=1..h}^{j=1..m}$

1. Put the small part of data aside as an evaluation set ES, keep the rest as a training set TS .
2. $\forall j\ \boldsymbol{c}_j(i) \leftarrow$ random sample from TS_1, $\forall j\ b_j(i), \Sigma_j^{-1}(i) \leftarrow$ small random value, $i \leftarrow 0$
3. $\forall j, p(i)$ in $\boldsymbol{c}_j(i), b_j(i), \Sigma_j^{-1}(i)$:
   $\Delta p(i) \leftarrow -\epsilon \frac{\delta E_1}{\delta p} + \alpha \Delta p(i-1), \qquad p(i) \leftarrow p(i) + \Delta p(i)$
4. $E_1 \leftarrow \sum_{\boldsymbol{x} \in TS_1} (f(\boldsymbol{x}) - y_i)^2$, $E_2 \leftarrow \sum_{\boldsymbol{x} \in TS_2} (f(\boldsymbol{x}) - y_i)^2$
5. If E_1 and E_2 are decreasing, $i \leftarrow i+1$, go to 3, else STOP. If E_2 started to increase, STOP.

Fig. 3. Gradient learning of RBF network

Evolutionary algorithms have been used as a very robust and efficient search procedure in various tasks, including neural networks training [7,8]. In [6] we have shown that in the case of RBF networks, the evolutionary learning has both certain advantages and drawbacks. On one hand, evolutionary algorithms are prone to local minima stucking problem, but on the other hand, they are less computationally efficient in finding best global solution. That is the main motivation behind hybrid approaches that combine the evolutionary framework with local fine tuning operations, such as various gradient algorithms.

Our proposed memetic algorithm (see Fig. 4) works with the population of *individuals* which represent encoded parameterizations of regularization networks.

By an *individual* I we understand a vector of floating-point encoded values $(c_{11}, \ldots, c_{1n}\ b_1, w_1, \ldots c_{h1}, \ldots, c_{hn} b_h, w_h)$, where h is a number of units, n is an input dimension, $w_i \in R$ are weights of linear combination, $b_i \in R$ are widths, and $c_{ij} \in R$ are centroid positions of every unit.

Such an individual I corresponds to a regularization network computing function $F_I(x) : R^n \rightarrow R$

$$F_I(x) = \sum_{i=1}^{h} w_i K_{b_i}(c_i, x)$$

The *fitness* f_I of an individual I is computed by means of an performance error over a given training set $\{(x_1, d_1), \ldots (x_T, d_T)\}$, e.g.

$$f_I = 100 - 100 \frac{1}{T} \sum_{t=1}^{T} \left(F_I(x_t) - d_t\right)^2 .$$

New populations are created using genetic operations of selection, mutation and crossover types. The selection operator is a fairly standard tournament selection with a small elitist rate.

There are three different operators of a mutation type: a small random change of parameterization, unit addition and a unit deletion. The first operator (cf. *mutate*) performs a small random change of individual parameters:

$$I_i = I_i + N(-1, 1),$$

The unit deletion operator (cf. *delete*) selects a substring of parameterization corresponding to randomly chosen regularization unit in a network, and deletes a this substring. The unit addition operator (cf. *insert*) randomly generates a substring of a size of one unit in a regularization network, and inserts it into a random position on the unit borders of the original parameterization. It is clear to see that the last two operations alter the network size in a opposite manner.

The crossover operator is a simple 1-point crossover on a floating point vector.

Before the fitness is evaluated, the gradient local search (cf. *gradient*) for an individual I performs n_g steps of gradient algorithm starting from the current parameterization value I. It uses the error partial derivative values to compute the gradient and then perform a small change of parameters accordingly (for details cf. [6])

$$I_i = I_i - \alpha \frac{\partial E}{\partial I_i}$$

where $\alpha > 0$ is a learning rate parameter.

The whole scheme of the memetic algorithm is sketched in Fig. 4 The starting population consists of networks with one unit only letting the evolution freedom to create bigger networks by mutation operator.

The second algorithm—referred to as a hybrid one—proposed in this paper is a combination of evolutionary learning described above that sets the centers and widths of the network, followed by a linear gradient learning to set the output weights. The advantage, as opposed to the gradient algorithm described earlier is that the procedure is

1. Create randomly an initial population P_0 of M individuals.
2. $i \leftarrow 0$
3. For each individual $I \in P_i$:
   (a) $gradient(I)$: perform local search by gradient descent algorithm.
   (b) evaluate fitness f_I.
4. $P_{i+1} \leftarrow$ empty set
5. $I_1 \leftarrow selection(P_i); I_2 \leftarrow selection(P_i)$
6. with probability p_{cross}: $(I_1, I_2) \leftarrow crossover(I_1, I_2)$
7. with probability p_{mutate}: $I_k \leftarrow mutate(I_k), k = 1, 2$
8. with probability p_{delete}: $I_k \leftarrow delete(I_k), k = 1, 2$
9. with probability p_{insert}: $I_k \leftarrow insert(I_k), k = 1, 2$
10. insert I_1, I_2 into P_{i+1}
11. if P_{i+1} has less then M individuals goto 5
12. $i \leftarrow i + 1$
13. goto 4 and iterate until the fitness stops increasing

Fig. 4. Memetic learning algorithm

1. Create randomly an initial population P_0 of M individuals.
2. $i \leftarrow 0$
3. For each individual I:
   (a) solve linear equations for w_{Ih}, and fixed them
   (b) evaluate the fitness of I
4. $P_{i+1} \leftarrow$ empty set
5. $I_1 \leftarrow selection(P_i); I_2 \leftarrow selection(P_i)$
6. with probability p_{cross}: $(I_1, I_2) \leftarrow crossover(I_1, I_2)$
7. with probability p_{mutate}: $I_k \leftarrow mutate(I_k), k = 1, 2$
8. with probability p_{delete}: $I_k \leftarrow delete(I_k), k = 1, 2$
9. with probability p_{insert}: $I_k \leftarrow insert(I_k), k = 1, 2$
10. insert I_1, I_2 into P_{i+1}
11. if P_{i+1} has less then M individuals goto 5
12. $i \leftarrow i + 1$
13. goto 4 and iterate until the fitness stops increasing

Fig. 5. Memetic learning algorithm

linear, using any matrix pseudo inverse method with good numerical properties (such as QR decomposition). Thus, we aim for the simplicity of the original regularization network algorithm by combining the evolutionary learning (for hidden layer) with linear optimization (for output layer). The scheme is sketched in the Fig. 5. Note, that the *individual* for the algorithm does not evolve all the parameters from the network, the w_h weights are set by linear optimization and remain fixed for mutation and crossover in one generation.

4 Experiments

The memetic and hybrid algorithm was tested on tasks from Proben1 data repository [9], which contains benchmark data sets used for neural network testing. The algorithms were run 10 times and average performance was evaluated.

Table 1 lists error on training and testing set achieved by our two proposed algorithms, and by the original regularization network learning algorithm. In the terms of training errors the memetic algorithm performed better in 4 cases, in the rest 5 cases the error is slightly higher than for regularization networks. The hybrid algorithm performed six times better w.r.t. the training error. Regarding the generalization capability, i.e. the testing error, the memetic algorithm performed better in 6 cases, while the hybrid never won, although the results are within the same order. Note that the number of hidden units needed by regularization networks is much higher than the ones needed by any of our two hybrid algorithms.

Figure 6 a) shows the flow of error function during the run of the memetic algorithm, while b) demonstrates an evolution of the number of hidden units. Figure 7 shows fitness and error function evolution for the hybrid algorithm on a two-dimensional test task.

Table 1. Comparison of hybrid, memetic, and original RN learning algorithms

	Hybrid alg. 2			Memetic alg.			RN		
Task	E_{train}	E_{test}	units	E_{train}	E_{test}	units	E_{train}	E_{test}	units
cancer1	1.45	2.43	8	1.85	1.49	8	2.28	1.75	525
cancer2	1.67	3.77	8	1.68	2.97	8	1.86	3.01	525
cancer3	1.63	4.10	8	1.77	2.78	8	2.11	2.79	525
card1	9.62	10.74	8	8.88	9.82	9	8.75	10.01	518
card2	8.43	15.18	8	7.15	13.39	15	7.55	12.53	518
card3	8.98	12.58	8	8.14	12.48	13	6.52	12.35	518
diabetes1	13.89	17.18	8	14.52	16.01	7	13.97	16.02	576
diabetes2	13.33	19.76	8	14.40	18.02	5	14.00	16.77	576
diabetes3	13.78	18.07	8	14.86	15.33	5	13.69	16.01	576

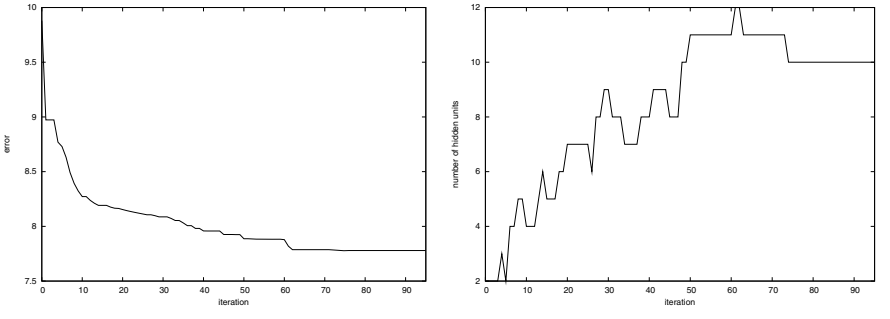


Fig. 6. a) Graph of a fitness function evolution. b) Number of hidden units during evolution.

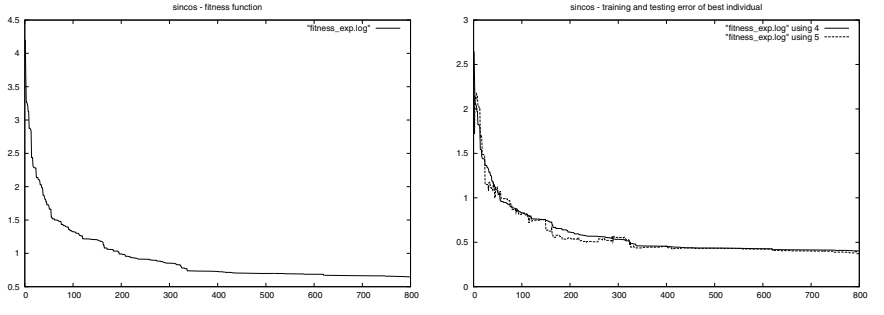


Fig. 7. a) Graph of a fitness function evolution for the second algorithm and sin×cos function. b) Training and testing error values for the same task.

5 Conclusion

Most of the learning algorithms work with networks of fixed architectures. Those optimizing also the number of hidden units can be divided into two groups – incremental and pruning. Pruning algorithm starts with large networks and tries to eliminate the irrelevant units, while incremental algorithms start with small network and add units as long as the network performance improves. The mentioned theoretical results speaks in favor of incremental algorithms. First, learning of small networks is fast since small numbers of parameters has to be optimized. Second, it is quite probable that reasonable solution will be found among smaller networks. Based on our experiments, we recommend to start with small number of hidden units and increase the network size only as long as also generalization ability improves.

Two hybrid learning algorithms were proposed, combining global approximation by evolutionary algorithm and local search by gradient descent, or linear optimization, respectively. Based on the mentioned results, both algorithms work as incremental. The performance of the memetic and hybrid algorithm was demonstrated on experiments, and it was shown that they both achieve the solution comparable with the full regularization network with much smaller number of hidden units.

Acknowledgments

This research has been supported by the Grant Agency of the Czech Republic under project no. 201/08/1744.

References

1. Haykin, S.: Neural Networks: a comprehensive foundation, 2nd edn. Tom Robins (1999)
2. Vidnerova, P., Neruda, R.: Testing error estimates for regularization and radial function networks. In: Proceedings of the ISNN 2009. LNCS. Springer, Heidelberg (2009)
3. Tikhonov, A., Arsenin, V.: Solutions of Ill-posed Problems. W.H. Winston, Washington (1977)
4. Poggio, T., Girosi, F.: A theory of networks for approximation and learning. Technical report, Cambridge, MA, USA, A. I. Memo No. 1140, C.B.I.P. Paper No. 31 (1989)
5. Poggio, T., Smale, S.: The mathematics of learning: Dealing with data. Notices of the AMS 50, 536–544 (2003)
6. Neruda, R., Kudová, P.: Learning methods for radial basis functions networks. Future Generation Computer Systems 21, 1131–1142 (2005)
7. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 9(87), 1423–1447 (1999)
8. Stanley, K.O., D'Ambrosio, D., Gauci, J.: A hypercube-based indirect encoding for evolving large-scale neural networks. Artificial Life 15(2) (2009)
9. Prechelt, L.: PROBEN1 – a set of benchmarks and benchmarking rules for neural network training algorithms. Technical Report 21/94, Universitaet Karlsruhe (September 1994)

Synchronization for a Class of Uncertain Chaotic Cellular Neural Networks with Time-Varying Delay

Jianjun Tu and Hanlin He

College of Sciences, Naval University of Engineering,
Wuhan 430033, China

Abstract. Synchronization problem of a class of chaotic cellular neural networks with time-varying delay and parameter uncertainties is probed in. Given the Lipschitz conditions, based on Lyapunov-Krasovskii functional method and the analysis of the bounded uncertainties, a kind of convenient linear controller is derived. It can be designed by only solving a linear matrix inequality and can lead the error system to be globally asymptotical stable. The simulation results indicate the super properties of the controller.

Keywords: Chaos, Cellular neural networks, Synchronization, Parameter uncertainty, Time delay, Lyapunov-Krasovskii functional.

1 Introduction

Cellular neural network (CNN) is a kind of nonlinear processors array, whose powerful functionality and simple implementation endow it broad applications. Besides software implementation, its hardware constructions have been gradually achieved. In [1], a kind of spatial domain equivalent framework of the sigma-delta (SD) modulation by the CNN is composed to reconstruct the 2D image and shows excellent reconstruction capabilities. Then, in [2], a low cost hardware implementable SD-CNN is proposed. In [3], a new design procedure for synthesizing associative memories based on a novel CNN structure is presented. The designed parameters are few, even though the dimension of desired patterns may be very high.

As an important issue of the control theory, chaos synchronization has been probed in by many scholars. Many kinds of synchronized methods have been put forward and intensively studied, such as sliding mode control method [4], robust control method [5], fuzzy logic control method [6] or adaptive control method [7]. Its theories have also been applied in many fields of engineering, such as medical science [8], laser control [9], secret communication [10], energy science [11], et al. Under certain conditions, CNN can presents chaotic behavior of three or even higher dimensions [12], which gives more possibilities to be used in abovementioned applications. One example is that a secure communication system based on the hyperchaotic synchronization of the quantum cellular neural network is presented in [13]. However, due to the effects of complex environmental

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 542–547, 2010.

factors, we should take the time delays and uncertainties into consideration. In this paper, bounded parameter distortions and time-varying delays are brought into the driving and responsive cellular neural networks, and a globally asymptotical synchronous controller is designed, which is linear and convenient to get, by only solving a linear matrix inequality.

2 Design of the Synchronization Controller

Choose the controlled response system and the driving system as following cellular neural networks with time-varying delays:

$$\dot{x} = (-C - \Delta C)\, x(t) + (A + \Delta A)\, f(x(t)) + (B + \Delta B)\, f(x\,(t - \tau\,(t))) + u, \quad (1)$$

$$\dot{y} = (-C - \Delta C)\, y(t) + (A + \Delta A)\, f(y(t)) + (B + \Delta B)\, f(y\,(t - \tau\,(t))), \quad (2)$$

where $x(t), y(t), u(t) \in \mathbf{R}^n$, $0 < \tau(t) < \infty$, $\dot{\tau}(t) \le \alpha < 1$, $f(x(t)) = (f_1(x_1), \cdots, f_n(x_n))^T$, $f(y(t)) = (f_1(y_1), \cdots, f_n(y_n))^T$, and $f_i(x_i(t)) = 0.5\, l_i(|x_i + 1| - |x_i - 1|)$. Hence the Lipschitz constants is

$$0 \le \frac{f_i(\xi_1) - f_i(\xi_2)}{\xi_1 - \xi_2} \le l_i, \quad i = 1, 2, \cdots, n. \quad (3)$$

Denote ΔC, ΔA and ΔB as the uncertainties, which can be described as $\Delta C = H_1 F_1 E_1$, $\Delta A = H_2 F_2 E_2$, $\Delta B = H_3 F_3 E_3$. $H_i, E_i, i = 1, 2, 3$ are real constant matrices with appropriate dimensions, and $F_i(t), i = 1, 2, 3$ are uncertain matrices, which satisfy $F_i^T F_i \le I$, where I is the identity matrix. Define $e(t) = x(t) - y(t)$, then the error system is

$$\dot{e}(t) = -\,(C + \Delta C)\, e(t) + (A + \Delta A)\, g(e(t)) + (B + \Delta B)\, g(e(t - \tau(t))) + u, \quad (4)$$

in which $g\big(e(t)\big) = f\big(e(t) + y(t)\big) - f\big(y(t)\big)$. We know that $e(t) = 0$ is the fixed point of $g\big(e(t)\big)$. According to (3), $g\big(e(t)\big)$ satisfies

$$0 \le \frac{g_i\big(e(t)\big)}{e_i(t)} \le l_i, \quad i = 1, 2, \cdots, n. \quad (5)$$

Lemma 1 [14]**.** Let H, F, E be real matrices of appropriate dimensions with $F^T F \le I$. Then for any scalar $\varepsilon > 0$, $HFE + E^T F^T H^T \le \varepsilon^{-1} H H^T + \varepsilon E^T E$.

According to lemma 1, we have

$$\begin{cases} -E_1^T F_1^T H_1^T P - P H_1 F_1 E_1 \le \varepsilon_1^{-1} P H_1 H_1^T P + \varepsilon_1 E_1^T E_1, \\ A E_2^T F_2^T H_2^T + H_2 F_2 E_2 A^T \le \varepsilon_2^{-1} H_2 H_2^T + \varepsilon_2 A E_2^T E_1 A^T, \\ B E_3^T F_3^T H_3^T + H_3 F_3 E_3 B^T \le \varepsilon_3^{-1} H_3 H_3^T + \varepsilon_3 B E_3^T E_3 B^T. \end{cases} \quad (6)$$

Theorem 1. Consider (1) and (2), the globally asymptotical synchronization between them can be achieved under the control

$$u = -\frac{1}{2}\big(AA^T + BB^T + \sum_{i=1}^{3} \varepsilon_i^{-1} H_i H_i^T + \varepsilon_2 A E_2^T E_2 A^T + \varepsilon_3 B E_3^T E_3 B^T + \sum_{i=2}^{3} \|E_i\|^2 \|H_i\|^2 I\big) P e(t), \tag{7}$$

where $\varepsilon_1, \varepsilon_2, \varepsilon_3$ are the parameters satisfying (6), $\|\cdot\|$ is the 2-norm of the matrix. P is the positive definite symmetric solution of the following linear inequality

$$C^T + PC - \frac{2-\alpha}{1-\alpha} L - \varepsilon_1 E_1^T E_1 > 0. \tag{8}$$

Proof: Choose a positive definite functional $V\big(t, \tau(t)\big) = V_1(t) + V_2\big(t, \tau(t)\big)$ as Lyapunov functional, where

$$\begin{cases} V_1(t) = e^T(t) P e(t), \\ V_2\big(t, \tau(t)\big) = \frac{1}{1-\alpha} \int_{t-\tau(t)}^{t} e^T(\theta) L e(\theta) d\theta, \end{cases} \tag{9}$$

in which P and L are both positive definite matrices. Take the derivative of $V_1(t)$ along (4), we have

$$\begin{aligned} \dot{V}_1(t) &= -e^T(t) \left[(C+\Delta C)^T P + P(C+\Delta C)\right] e(t) + 2g^T\big(e(t)\big)(A+\Delta A)^T P e(t) \\ &\quad + 2g^T\big(e(t-\tau(t))\big)(B+\Delta B)^T P e(t) + 2e^T(t) P u \\ &\le -e^T(t) \left[(C+\Delta C)^T P + P(C+\Delta C)\right] e(t) + \sum_{i=1}^{n} g_i^2\big(e(t)\big) \\ &\quad + e^T(t) P (A+\Delta A)(A+\Delta A)^T P e(t) + \sum_{i=1}^{n} g_i^2\big(e(t-\tau(t))\big) \\ &\quad + e^T(t) P (B+\Delta B)(B+\Delta B)^T P e(t) + 2e^T(t) P u \\ &\le -e^T(t) \left[(C+\Delta C)^T P + P(C+\Delta C)\right] e(t) + e^T(t) L e(t) \\ &\quad + e^T(t) P (A+\Delta A)(A+\Delta A)^T P e(t) + e^T(t-\tau(t)) L e(t-\tau(t)) \\ &\quad + e^T(t) P (B+\Delta B)(B+\Delta B)^T P e(t) + 2e^T(t) P u, \end{aligned} \tag{10}$$

where $L = \operatorname{diag}\big(l_1^2, l_2^2, \cdots, l_n^2\big)$. Owing to following inequalities

$$\begin{aligned} (A+\Delta A)(A+\Delta A)^T &= AA^T + A\Delta A^T + \Delta A \Delta A^T \\ &\le AA^T + \varepsilon_2^{-1} H_2 H_2^T + \varepsilon_2 A E_2^T E_2 A^T \\ &\quad + \|E_2\|^2 \|H_2\|^2 I, \end{aligned} \tag{11}$$

$$\begin{aligned} (B+\Delta B)(B+\Delta B)^T &= BB^T + B\Delta B^T + \Delta B \Delta B^T \\ &\le BB^T + \varepsilon_3^{-1} H_3 H_3^T + \varepsilon_3 B E_3^T E_3 B^T \\ &\quad + \|E_3\|^2 \|H_3\|^2 I, \end{aligned} \tag{12}$$

$$-\Delta C^T P - P \Delta C \le \varepsilon_1^{-1} P H_1 H_1^T P + \varepsilon_1 E_1^T E_1, \tag{13}$$

which are in accordance with (6), we have

$$\dot{V}_1 \le e^T(t)\Big[- C^T P - PC + L + P(AA^T + BB^T)P + \sum_{i=1}^{3} \varepsilon_i^{-1} P H_i H_i^T P$$
$$+P(\varepsilon_2 A E_2^T E_2 A^T + \varepsilon_3 B E_3^T E_3 B^T)P + \sum_{i=2}^{3} \|E_i\|^2 \|H_i\|^2 PP + \varepsilon_1 E_1^T E_1 \Big] e(t)$$
$$+e^T(t-\tau(t))Le(t-\tau(t)) + 2e^T Pu \tag{14}$$

and

$$\dot{V} = \dot{V}_1 + \dot{V}_2 \le e^T(t)\Big[- C^T P - PC + \frac{2-\alpha}{1-\alpha} L + P(AA^T + BB^B)P$$
$$+ \sum_{i=1}^{3} \varepsilon_i^{-1} P H_i H_i^T P + P(\varepsilon_2 A E_2^T E_2 A^T + \varepsilon_3 B E_3^T E_3 B^T)P$$
$$+ \sum_{i=2}^{3} \|E_i\|^2 \|H_i\|^2 PP + \varepsilon_1 E_1^T E_1 \Big] e(t) + 2e^T(t) Pu. \tag{15}$$

Then, choose (7) as the controller, we can get $\dot{V} \le -e^T(t) Q e(t)$, where

$$Q = C^T P + PC - \frac{2-\alpha}{1-\alpha} L - \varepsilon_1 E_1^T E_1. \tag{16}$$

So if $C^T P + PC - \frac{(2-\alpha)L}{1-\alpha} - \varepsilon_1 E_1^T E_1 > 0$, a constant $\rho > 0$ can be found out to make $\dot{V} \le -\rho \|e(t)\|^2$. Then the controller will offer the error system globally asymptotical stability. This completes the proof.

3 Example and Simulations

Consider a class of uncertain chaotic cellular neural networks with time-varying delays as the driving system and the response system, with the system matrices and structure expressions of (1) and (2) assigned as

$$x = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}, y = \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix}, f\big(x(t)\big) = \begin{pmatrix} 0.5\big(|x_1+1| - |x_1-1|\big) \\ 0.5\big(|x_2+1| - |x_2-1|\big) \end{pmatrix}, C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

$$A = \begin{pmatrix} \frac{\pi+4}{4} & 20 \\ 0.1 & \frac{\pi+4}{4} \end{pmatrix}, B = \begin{pmatrix} \frac{-\sqrt{2}\cdot 1.3\pi}{4} & 0.1 \\ 0.1 & \frac{-\sqrt{2}\cdot 1.3\pi}{4} \end{pmatrix}, H_1 = E_1 = \begin{pmatrix} 0.4 & 0 \\ 0 & -0.4 \end{pmatrix}, \tag{17}$$

$H_2 = E_2 = \mathrm{diag}(0, 0.2)$, $H_3 = E_3 = \mathrm{diag}(0.2, 0)$. Note that $l_1 = l_2 = 1$. The chaotic attractor of the driving system is shown in Fig. 1.

In order to use theorem 1, we should find a solution P of (8). With ε_i, $i = 1, 2, 3$ chosen as 1, and α chosen as 0.9, a solution is

$$P = \begin{pmatrix} 7.5 & 0 \\ 0 & 7.5 \end{pmatrix}. \tag{18}$$

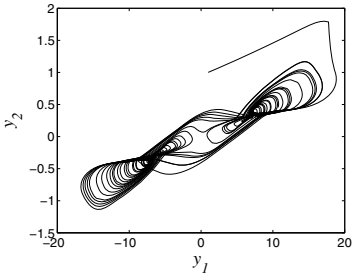


Fig. 1. The chaotic attractor of the driving system

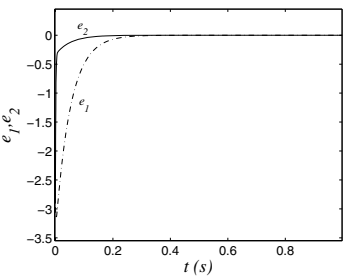


Fig. 2. Synchronization errors of the driving and respond systems

The synchronous controller can be derived as

$$u = \begin{pmatrix} -1581 & -139 \\ -139 & -21 \end{pmatrix} e(t). \tag{19}$$

With the initial conditions of the driving system and the response system given as $y(0) = [3 \quad 3]^T$ and $x(0) = [0.1 \quad 0.1]^T$, the synchronization errors are shown in Fig. 2.

As seen in Fig. 2, the synchronous error curves can convergence to 0 within 0.4s without any overshoot, which reflects the rapidity of the synchronous controller.

4 Conclusions

In this paper, the synchronization problem of a class of uncertain chaotic cellular neural networks with variable time delays is probed in, and a kind of convenient linear controller is derived. Moreover, the problem with the piecewise links of the cellular neural networks can be effectively solved by this method. This set of theoretical methods can be used in more widespread synchronization problems of time-delayed and uncertain systems.

Acknowledgments. The authors acknowledge the support of the National Natural Science Foundation of China (Grant No. 60974136).

References

1. Hisashi, A., Tsuyoshi, O., Nobuaki, T., Mamoru, T.: Sigma-Delta Cellular Neural Network for 2D Modulation. Neural Networks 21(2-3), 349–357 (2008)
2. Aomori, H., Naito, Y., Otake, T., Takahashi, N., Matsuda, I., Itoh, S., Tanaka, M.: Hardware-Aware Model of Sigma-Delta Cellular Neural Network. In: European Conference on Circuit Theory and Design, pp. 311–314. IEEE Press, New York (2009)
3. Zeng, Z.G., Wang, J.: Associative Memories based on Continuous-Time Cellular Neural Networks Designed using Space-Invariant Cloning Templates. Neural Networks 22(5-6), 651–657 (2009)

4. Mohammad, S.T., Mohammad, H.: Synchronization of Chaotic Fractional-Order Systems via Active Sliding Mode Controller. Physica A 387(1), 57–70 (2008)
5. Cheng, C.K., Kuo, H.H., Hou, Y.Y., Hwang, C.C., Liao, T.L.: Robust Chaos Synchronization of Noise-Perturbed Chaotic Systems with Multiple Time-Delays. Physica A 387(13), 3093–3102 (2008)
6. Yau, H.T., Shieh, C.S.: Chaos Synchronization using Fuzzy Logic Controller. Nonlinear Analysis: Real World Applications 9(4), 1800–1810 (2008)
7. Lin, W.: Adaptive Chaos Control and Synchronization in only Locally Lipschitz Systems. Physics Letters A 372(18), 3195–3200 (2008)
8. Sato, D., Xie, L.H., Sovari, A.A., Tran, D.X., Morita, N., Xie, F.G., Karagueuzian, H., Garfinkel, A., Weiss, J.N., Qu, Z.L.: Chaos Synchronization in the Genesis of Cardiac Arrhythmias. Biophysical Journal 96(3), 666a (2009)
9. Cheng, D.L., Huang, C.F., Cheng, S.Y., Yan, J.J.: Synchronization of Optical Chaos in Vertical-Cavity Surface-Emitting Lasers via Optimal PI Controller. Expert Systems with Applications 36(3), 6854–6858 (2009)
10. Kaddoum, G., Roviras, D., Charg, P., Danile, F.P.: Robust Synchronization for Asynchronous Multi-User Chaos-based DS-CDMA. Signal Processing 89(5), 807–818 (2009)
11. Li, X.C., Xu, W., Li, R.H.: Chaos Synchronization of the Energy Resource System. Chaos, Solitons & Fractals 40(2), 642–652 (2009)
12. Wang, X.Y., Xu, B., Zhang, H.G.: A Multiary Number Communication System based on Hyperchaotic System of 6th-Order Cellular Neural Network. Communications in Nonlinear Science and Numerical Simulation 15(1), 124–133 (2010)
13. Wang, S., Cai, L., Kang, Q., Li, Q., Wu, G.: Secure Communication based on Tracking Control of Quantum Cellular Neural Network. In: Proceedings of the 7th World Congress on Intelligent Control and Automation, pp. 8059–8063. IEEE Press, Piscataway (2008)
14. Chen, W.-H., Guan, Z.-H., Lu, X.-M., Yang, X.-F.: Delay-Dependent Absolute Stability of Uncertain Lur'e System with Time-Delays. Acta automatica sinica 30(2), 235–238 (2004)

Global Exponential Stability of Equilibrium Point of Hopfield Neural Network with Delay

Xiaolin Liu[1] and Kun Yuan[2]

[1] College of Aeronautical Automation, Civil Aviation University of China, TianJin 300300, China
[2] College of Astronautics, Northwestern Polytechnical University, XiAn 710072, China
missxiaolin1997@163.com

Abstract. In this paper, the global exponential stability of the Hopfield neural network with delay is studied. By using of the methods of constant variation and variable substitution, a new sufficient global exponential stable criterion for the equilibrium point of the network is derived. The result is different to the known references and is realizable easily.

Keywords: Delay, Hopfield neural network, Equilibrium point, Global exponential stability.

1 Introduction

Since the American biologist Hopfield brought forward the Hopfield neural network in 1982, it has attracted much attention for the wide applications in many areas such as associative memory, pattern recognition and optimization in recent years, because of its rich dynamic behavior and overall computing power, which is pointed out in [1]-[3]. However, a number of issues in the hardware implementation of the neural network, such as switching delay and communication delay, greatly reduce the dynamic performance of the hardware neural network and may lead to network instability, because one of the key factors is the network time delay, which is described in [4]-[6]. Therefore, it is very important and significant to study the stability of the Hopfield neural network.

2 Preliminary

The Hopfield neural network with delay can be described by the following differential equations

$$\frac{du_i(t)}{dt} = -b_i u_i(t) + \sum_{j=1}^{n} T_{ij} f_j(a_j u_j(t-\tau_j)) + I_i \ , b_i > 0, i = 1, 2, ..., n \ . \tag{1}$$

where n is the quantity of the neuron in the network, $u_j(t)$ denotes the state variable of the j-th neuron at time t, $f_j(u_j(t-\tau_j))$ denotes the output of the j-th neuron at

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 548–553, 2010.

time $t-\iota_j$, b_i, T_{ij}, a_j, I_i are constants, b_i denotes the rate with which the i-th unit will resets its potential to the resting state in isolation when disconnected form the network and external inputs, T_{ij} denotes the influence intensity of the output of the j-th neuron toward the i-th neuron at time t, I_i is the input, ι_j denotes the transfer delay of the j-th neuron.

According to [7] and [8], the initial conditions associated with system (1) are of the following form

$$u_i(t) = \phi_i(t), t \in [t_0 - \iota, t_0], \iota = \max_{1 \le i \le n} \iota_i, i = 1, 2, ..., n .$$

where $\phi_i(t)$ are bounded continuous functions defined on $[t_0 - \iota, t_0]$.

Assume that the activation functions $f_j (j = 1, 2, ..., n)$ are continuous and can be characterized by the following assumptions.

(H_1) f_j are bounded defined on R.

(H_2) There exist constants $\mu_j > 0$, such that f_j satisfies $|f_j(x) - f_j(y)| \le \mu_j |x - y|$ for all $x, y \in R$ and $i = 1, 2, ..., n$.

The following lemmas are given to assure the existence of an equilibrium point and its global exponential stability.

Lemma 1. If $f_j (j = 1, 2, ..., n)$ satisfy the assumptions (H_1) and (H_2), then there exist an equilibrium point of system (1).

Proof. If $u^* = (u_1^*, u_2^*, ..., u_n^*)^T$ is an equilibrium point of system (1), then u^* satisfies the following nonlinear equation

$$u_i^* = \frac{1}{b_i}\left[\sum_{j=1}^{n} T_{ij} f_j(a_j u_j^*) + I_i\right] = \sum_{j=1}^{n} \frac{T_{ij} f_j(a_j u_j^*)}{b_i} + \frac{I_i}{b_i}$$

$$= \sum b_{ij} f_j(a_j u_j^*) + q_i \quad i = 1, 2, ..., n . \tag{2}$$

where $B = (b_{ij})_{n \times n} = \left| T_{ij} / b_i \right|_{n \times n}$, $q = (q_1, q_2, ..., q_n)^T$, $q_i = I_1 / b_i$, $i = 1, 2, ..., n$. Therefore, (2) can be rewritten in the following vector- matrix form

$$u^* = F(u^*) = Bf(u^*) + q . \tag{3}$$

where $f(u^*) = (f_1(a_1 u_1^*), f_2(a_2 u_2^*), ..., f_n(a_n u_n^*))^T$, u^* is an fixed point for $F : R^n \to R^n$. The existence of the fixed point of the mapping F can be proved by the Brouwer fixed point theorem. In fact, a hypercube is defined by

$$\Omega = \left| u \in R^n \middle| \, \|u - q\|_\infty \le \|B\|_\infty M \right| . \tag{4}$$

where $M = \max_i \sup_s |f_i(s)|$ .Then

$$\|F(u)-q\|_\infty = \|Bf(u)\|_\infty \le \|B\|_\infty \|f(u)\|_\infty \le \|B\|_\infty M . \tag{5}$$

where the mapping F is continuous and F maps a closed convex set into itself. Consequently, based on the Brouwer fixed point theorem, F has at least one fixed point. This means that there exists at least one equilibrium point $u^* = (u_1^*, u_2^*, ..., u_n^*)^T$ for system (1).

Based on the above analysis, if $f_j(j = 1, 2, ..., n)$ satisfy the assumptions (H_1) and (H_2), system (1) exists an equilibrium point. The proof is completed.

Lemma 2. If $f_j(j = 1, 2, ..., n)$ satisfy the assumptions (H_1) and (H_2), then any solution of system (1) is bounded on $[0, \infty)$.

Proof. Any solution of (1) satisfies the following differential inequality

$$-b_i u_i(t) - \beta_i \le \frac{du_i(t)}{dt} \le -b_i u_i(t) + \beta_i . \tag{6}$$

where $\beta_i = \sum_{j=1}^{n} (|T_{ij}|) \sup_{s \in R} |f_i(s)| + I_i$. Therefore, any solution of (1) is bounded on $[0, \infty)$. The proof is completed.

3 Stability Researches

Theorem. If $f_j(j = 1, 2, ..., n)$ satisfy the assumptions (H_1) and (H_2), the parameters μ_j , a_j , T_{ij} , $(i, j = 1, 2, ..., n)$ satisfies the following condition

$$\max_{1 \le i \le n} \left\{ \sum_{j=1}^{n} \mu_j |a_j| |T_{ij}| \right\} < 1 .$$

then the equilibrium point of system (1) is global exponential stable.

Proof. If the Hopfield neural network converges to an equilibrium point, then the solution of (1) satisfies the following equation

$$\lim_{t \to \infty} \sum_{i=1}^{n} |u_i(t) - u_i^*| = 0 . \tag{7}$$

By using the method of variable substitution, set $y_i(t) = u_i(t) - u_i^*$, $i = 1, 2, ..., n$, then

$$\frac{dy_i(t)}{dt} = -b_i y_i(t) + \sum_{j=1}^{n} T_{ij} a_j (f_j(u_j^* + y_j(t - \iota_j)) - f_j(u_j^*)) . \tag{8}$$

Therefore, in order to study the global exponential stability of the equilibrium point of system (1), it is equivalent to study the global exponential stability of the zero solution of (8). The research method of the neural network stability is simplified.

By using the method of constant variation, (9) can be deduced.

$$y_i(t) = b_i y_i(t_0) e^{-(t-t_0)} + \sum_{j=1}^{n} \int_{t_0}^{t} e^{-(t-s)} \left[T_{ij} a_j (f_j(u_j^* + y_j(s-\iota_j)) - f_j(u_j^*)) \right] ds . \tag{9}$$

Considering the assumption (H_2), (10) can be deduced.

$$\begin{aligned} |y_i(t)| &\le b_i |y_i(t_0)| e^{-(t-t_0)} + \sum_{j=1}^{n} \int_{t_0}^{t} e^{-(t-s)} \left[|T_{ij}| |a_j| |f_j(u_j^* + y_j(s-\iota_j)) - f_j(u_j^*)| \right] ds \\ &\le b_i \phi e^{-(t-t_0)} + \sum_{j=1}^{n} \mu_j |a_j| \int_{t_0}^{t} e^{-(t-s)} |T_{ij}| |y_j(s-\iota_j)| ds . \end{aligned} \tag{10}$$

When $t > t_0$, set $P_i(t) = b_i \phi e^{-(t-t_0)} + \sum_{j=1}^{n} \mu_j |a_j| \int_{t_0}^{t} e^{-(t-s)} |T_{ij}| |y_j(s-\iota_j)| ds$. When $t_0 - \iota \le t \le t_0$, set $P_i(t) = b_i \phi$.

Considering the assumption (H_2), when $t \ge t_0 - \iota$, (11) can be deduced.

$$|y_i(t)| \le P_i(t), i = 1, 2, ..., n . \tag{11}$$

Considering the initial conditions, when $t \ge t_0$, (12) can be deduced.

$$\begin{aligned} \frac{dP_i(t)}{dt} &= -b_i \phi e^{-(t-t_0)} - \sum_{j=1}^{n} \mu_j |a_j| \int_{t_0}^{t} e^{-(t-s)} |T_{ij}| |y_j(s-\iota_j)| ds + \sum_{j=1}^{n} \mu_j |a_j| |T_{ij}| |y_j(t-\iota_j)| \\ &\le -P_i(t) + \sum_{j=1}^{n} \mu_j |a_j| |T_{ij}| \sup_{-\iota \le \theta \le 0} P_j(t+\theta) . \end{aligned} \tag{12}$$

Set $S_i(t) = P_i(t) e^{\varepsilon(t+t-t_0)}, t \ge t_0 - \iota, i = 1, 2, ..., n$, then

$$\begin{aligned} \frac{dS_i(t)}{dt} &= P_i'(t) e^{\varepsilon(t+t-t_0)} + P_i(t) e^{\varepsilon(t+t-t_0)} \varepsilon \\ &\le \left[-P_i(t) + \sum_{j=1}^{n} \mu_j |a_j| |T_{ij}| \sup_{-\iota \le \theta \le 0} P_j(t+\theta) \right] e^{\varepsilon(t+t-t_0)} + P_i(t) e^{\varepsilon(t+t-t_0)} \varepsilon \\ &= (-1+\varepsilon) S_i(t) + \sum_{j=1}^{n} \mu_j |a_j| |T_{ij}| \sup_{-\iota \le \theta \le 0} S_j(t+\theta) . \end{aligned} \tag{13}$$

Considering (11), if the zero solution of system (9) is global exponential stable, then (14) can be deduced.

$$|y_i(t)| \le P_i(t) = S_i(t) e^{-\varepsilon(t+t-t_0)} \le \phi e^{-\varepsilon(t-t_0)}, t \ge t_0, i = 1, 2, ..., n . \tag{14}$$

Therefore, in order to study the global exponential stability of the zero solution of system (9), it is only to verify that there exists a sufficient small $\varepsilon > 0$, which satisfies the following equation

$$S_i(t) \le e^{\varepsilon t}\phi, t \ge t_0 - \iota . \tag{15}$$

A proof by contradiction is selected as the method to prove the above conclusion. Assume that (15) is not correct, there exist i and $t_1 > t_0$, which satisfy the following equation

$$S_i(t_1) = e^{\varepsilon t}\phi, S_i(t) < e^{\varepsilon t}\phi, S_j(t) \le e^{\varepsilon t}\phi, t_0 \le t \le t_1, j \ne i, j = 1,2,...,n . \tag{16}$$

In the one hand, (17) can be deduced.

$$\frac{dS_i(t_1)}{dt} \ge 0 . \tag{17}$$

In the other hand, because $\max\limits_{1\le i\le n}\left\{\sum\limits_{j=1}^{n}\mu_j\left|a_j\right|\left|T_{ij}\right|\right\} < 1$, (18) can be deduced.

$$-1+\varepsilon+\sum_{j=1}^{n}\mu_j\left|a_j\right|\left|T_{ij}\right| < 0, i = 1,2,...,n . \tag{18}$$

According to (13) and (18), (19) can be deduced.

$$\frac{dS_i(t_1)}{dt} \le (-1+\varepsilon)S_i(t_1) + \sum_{j=1}^{n}\mu_j\left|a_j\right|\left|T_{ij}\right| \sup_{-\iota\le\theta\le 0} S_j(t+\theta)$$

$$= (-1+\varepsilon)e^{\varepsilon t}\phi + \sum_{j=1}^{n}\mu_j\left|a_j\right|\left|T_{ij}\right|e^{\varepsilon t}\phi = \left[-1+\varepsilon+\sum_{j=1}^{n}\mu_j\left|a_j\right|\left|T_{ij}\right|\right]e^{\varepsilon t}\phi < 0 . \tag{19}$$

(19) is contradictory to $\frac{dS_i(t_1)}{dt} \ge 0$. Therefore, (15) is correct. So the zero solution of system (9) and the equilibrium point of system (1) are global exponential stable. The proof is completed.

4 Conclusions

In this paper, for the Hopfield neural network with delay, the existence of the equilibrium point and its global exponential stability are studied. Based on the methods of constant variation and variable substitution, the sufficient criterion is derived. The result is different to the known references and is realizable easily.

Acknowledgments. It is a project supported by the science and technology project of Civil Aviation University of China (05YK05M, ZXH2009D020).

References

1. Shao, J.-L., Huang, T.-Z.: Global exponential stability analysis of a class of dynamical neural networks. Journal of electronic science technology of China 7(2), 171–174 (2009)
2. Yan, Y.-L., Liao, W.-T.: Globally exponential stability of equilibrium for new Hopfield neural networks. Acta electronic sinica 36(8), 1543–1546 (2008)
3. Yang, F.-J., Zhang, C.-L., Wu, Q.-D.: Global exponential stability of Hopfield neural networks with delays. Pure and applied mathematics 24(1), 179–185 (2008)
4. Chen, W.-Y.: Globally exponential asymptotic stability of Hopfield neural network with time-varying delays. Acta scientiarum naturalium university Nankaiensis 38(5), 81–86 (2005)
5. Zhang, L.-J.: Global exponential stability of Hopfield neural networks with time-varying delay and the estimation of decay rate. Journal of northwest normal university (natural science) 43(2), 1–4 (2007)
6. Chen, W.-Y.: On the globally asymptotic stability for a class of Hopfield neural networks with Delay. Journal of biomathematics 19(2), 175–179 (2004)
7. Xu, J., Zhong, S.-M., Zhang, C.-F.: Stability of Hopfield type neural network with distributed time delay. Journal of UEST of China 33(2), 200–203 (2004)
8. Qian, Y., Feng, X.-M., Jiang, H.-J.: On Global Stability Criterion of High—order Hopfield Neural Networks with Distributed Delays. Journal of Xinjiang university (Natural science edition) 26(2), 53–59 (2009)

Stability of Impulsive Cohen-Grossberg Neural Networks with Delays*

Jianfu Yang[1], Wensi Ding[2], Fengjian Yang[1], Lishi Liang[3], and Qun Hong[4]

[1] Department of Computation Science, Zhongkai University of Agriculture and Engineering, Guangzhou, 510225, P.R. China
[2] School of Machanical and Automative Engineering of South China University of Technology, Guangzhou, 510640, P.R. China
[3] The 5th Middle School of Guangzhou City, Guangzhou, 510220
[4] The Haizhu District Chigang Primary School of Guangzhou City, Guangzhou, 510310
wensiding@126.com, yjf197202@163.com

Abstract. In this paper, with assuming global Lipschitz conditions on the activation functions, applying idea of vector Lyapunov function, Young inequality and Halanay differential inequality with delay, the global exponential stability of the equilibrium point for a class of Cohen-Grossberg neural networks with time-varying delays and impulses is investigated, the sufficient conditions for globally exponential stability of neural networks are obtained.

Keywords: global exponential stability; Lyapunov function.

1 Introduction

In recent decades, much attention has been devoted to the studies of the Cohen-Grossberg neural networks, there are many interesting phenomena in the dynamical behaviors of Cohen-Grossberg neural networks. The asymptotic stability, exponential stability, robust stability, periodic bifurcation and chaos of the neural network have been hot topics. Liu Jiang [1] studied the globally exponential stability of Cohen-Grossberg neural networks with time-varying delays described by the system of nonlinear delay differential equations:

$$\dot{u}_i(t)=-a_i(u_i(t))[b_i(u_i(t))-\sum_{j=1}^{n} T_{ij}f_j(u_j(t-\tau_j(t)))],(i,\ j=1,2,\cdots,n). \tag{1}$$

In this paper, we consider the globally exponential stability of following impulsive Cohen-Grossberg neural networks with time-varying delays:

$$\begin{cases}\dot{u}_i(t)=-a_i(u_i(t))[b_i(u_i(t))-\sum_{j=1}^{n} T_{ij}f_j(u_j(t-\tau_j(t)))],(i,\ j=1,2,\cdots,n).,t\neq t_k,t\geq 0,\\ \Delta u_i(t_k)=u_i(t_k^+)-u_i(t_k^-)=I_{ik}(u(t_k^-)),k\in N\\ u_i(s)=\phi_i(s),i=1,2,\cdots,n.\end{cases} \tag{2}$$

* This work was supported by the National Natural Science Foundation of China under Grants 50775075 and the State Key Laboratory Foundation under Grants VSN-2008-01.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 554–560, 2010.

In which the sequence of times $\{t_k\}_{k\in N}$ satisfies $t_1<t_2<\cdots<t_k<\cdots$ is a strictly increasing sequence such that $\lim_{k\to\infty} t_k=+\infty, u_i(t_k^-)$ and $u_i(t_k^+)$ denote the left-hand and right-hand limit at t_k, respectively, I_{ik} shows impulsive perturbation of the ith neuron at time t_k. We always assume

$$u_i(t_k)=u_i(t_k^-), k\in N \quad .$$

The system (2) is supplemented with initial values given by

$$u_i(t_0+t)=\varphi_i(t), t\in[-\tau,0], i=1,2,\cdots,n \ , \tag{3}$$

$\varphi_i(t)$ denote real-valued bounded and continuous functions defined on $[-\tau,0]$.

In this paper, we introduce the following assumptions for system (2):

H1) the activation function $f_j: R\to R$ is globally Lipschitz continuous with a Lipschitz constant $l_j>0$ namely $|f_j(u)-f_j(v)|\le l_j|u-v|$ for all $j=1,2,\cdots,n, u\ne v, u,v\in R$.

H2) Each function $a_i(u)$ is bounded, positive and locally Lipschitz continous. Furthermore, $0<\underline{a_i}\le a_i(u)\le\overline{a_i}<+\infty$ for all $u\in R=(-\infty,+\infty)$ and $i=1,2,\ldots,n$.

H3) Each function $b_i(u)$ is locally Lipschitz continous and $b_i(u)\in C^1(R,R)$ and there exists $\beta_i>0$ such that $b_i'(u)\ge\beta_i>0$ for $u\in R, i=1,2,\ldots,n$; both $b_i(\cdot)$ and $b_i^{-1}(\cdot)$ are locally Lipschitz continous.

2 Preliminaries

In the following, we will cite some definitions and some lemmas which will be used in the proof of the main results.

Definition 1. Vector function $u(t)=(u_1(t),\cdots,u_n(t))^T$ is called a solution of system (2), if $u(t)$ is continuous at $t\ne t_k$ and $t\ge 0, u(t_k)=u(t_k^-)$, and $u(t_k^+)$ exist, $u(t)$ satisfies Eq. (2) and initial condition (3). Especially, a point $u^*=(u_1^*,u_2^*,\cdots,u_n^*)$ in R^n is called an equilibrium point of (2), if $u(t)=u^*$ is a solution of (2). Where the components u_i^* are governed by the algebraic system

$$0=b_i(u_i^*)-\sum_{j=1}^{n} T_{ij} f_j(u_j^*) \tag{4}$$

Throughout this paper, we always assume that the impulsive jumps I_k satisfy $I_k(u_i^*)=0, i=1,\cdots,n, k\in N$. Here $u^*=(u_1^*,u_2^*,\cdots,u_n^*)^T$ is the equilibrium point of system (2). If $I_{ik}(u)=0$ for all $u\in R^n, i=1,\cdots,n, k\in N$, then the impulsive model (2) becomes continuous cellular neural networks with delays model (1). Let $u(t,t_0,\varphi)$ denote the solution of system (2) through (t_0,φ).

Definition 2. The equilibrium point $u^*=(u_1^*, u_2^*, \cdots, u_n^*)^T$ of system (2) is said to be globally exponential stable if there exist constants $\sigma>0$ such that $\|u(t)-u^*\| \leq \|\phi-u^*\|e^{-\sigma t}$ for all $t\geq 0$, where

$$\|\phi-u^*\|=\sup_{s\in[-\tau,0]}[\sum_{j=1}^{n}|\phi_j(s)-u_j^*|]\cdot$$

Definition 3. (Cao and Wang [2]) A map: $H:R^n\to R^n$ is a homeomorphism of R^n onto itself, if $H\in C^0$ is onto and the inverse map $H^{-1}\in C^0$.

Lemma 1. (Forti and Tesi [3]) If $H(x)\in C^0$ satisfies the following conditions:

(i) $H(x)$ is injective on R^n

(ii) $\|H(u)\|\to\infty$ as $\|u\|\to\infty$.

Then $H(x)$ is a homeomorphism of R^n onto itself.

Lemma 2. (Young inequality [4]) Assume that $\xi_1>0, \xi_2>0, l>1$ and $\frac{1}{l}+\frac{1}{m}=1$, then following inequality: $\xi_1\xi_2\leq\frac{1}{l}\xi_1^l+\frac{1}{m}\xi_2^m$ holds.

Lemma 3. (Halanay inequality [5]) Let α and β be constants with $\alpha>\beta>0$ and $u(t)$ be a nonnegative continuous function on $[t_0-\tau, t_0]$. If $u(t)$ satisfies the following inequality $u'(t)\leq-\alpha u(t)+\beta\overline{u}(t)$, where $\overline{u}(t)=\sup_{t-\tau\leq s\leq t}\{u(s)\}, \tau\geq 0$ is a constant, then for $t\geq t_0$, we have $u(t)\leq\overline{u}(t_0)e^{-r(t-t_0)}$. In which r is the unique positive solution of the equation $r=\alpha-\beta e^{r\tau}$.

3 Main Results

In this section, we will give the criteria to guarantee the globally exponential stability for the impulsive system (2). We consider the impulsive state displacements characterized by $I_k:R\to R$ at fixed instants of time $t=t_k, k\in N$ are defined by

$$u_i(t_k)-u_i(t_k^-)=I_k(u(t_k^-))=\gamma_{ik}(u(t_k^-)-u_i^*), k\in N, i=1,2,\cdots,n. \tag{5}$$

where γ_{ik} denote real numbers. For convenience in our analysis, we let

$$y_i(t)=u_i(t)-u_i^*, f_j(y_j(t-\tau_j(t)))=f_j(y_j(t-\tau_j(t))+u_j^*)-f_j(u_j^*), (j=1,2,\cdots,n).$$

So that (2) can be rewritten as

$$\begin{cases}\dot{y}_i(t)=-a_i(u_i(t))[b_i(u_i(t))-\sum_{j=1}^{n}T_{ij}f_j(u_j(t-\tau_j(t)))], t\geq 0,\\ y_i(t_k)=\gamma_{ik}y_i(t_k^-), i=1,2,\cdots,n; k\in N,\\ y_i(s)=\psi_i(s)=\phi_i(s)-u_i^*, s\in[-\infty,0].\end{cases} \tag{6}$$

Theorem 1. Let $T_{ij} \in R$, in addition to H1), H2), H3), suppose the condition:

H4) If
$$k_1=\min_{1\le i\le n}\{\gamma_i\}>\max_{1\le j\le n}\{\sum_{i=1}^{n} l_i|T_{ij}|\}=k_2>0 \tag{7}$$

is satisfied. Then there exists a unique equilibrium point u^* of (2).

Proof. Consider a map

$$H(u)=(H_1(u), H_2(u), \cdots, H_n(u))^T \in C^0(R^n, R^n)$$

where

$$H_i(u)=b_i(u_i)-\sum_{j=1}^{n} T_{ij} f_j(u_j), (i=1, 2, \cdots, n) \cdot$$

The map $H \in C^0$ is a homeomorphism on R^n if it is injective on R^n and satisfies $\|H(u)\| \to \infty$ as $\|u\| \to \infty$. We demonstrate the injective part, namely $H(u)=H(v)$ implies $u=v$ for any $u, v \in R^n$, as follows, we have

$$b_i(u_i)-\sum_{j=1}^{n} T_{ij} f_j(u_j) = b_i(v_i)-\sum_{j=1}^{n} T_{ij} f_j(v_j),$$

and consequently

$$|b_i(u_i)-b_i(v_i)| = \sum_{j=1}^{n} T_{ij} |f_j(u_j)-f_j(v_j)| ,$$

which leads to

$$\sum_{i=1}^{n} \gamma_i |u_i-v_i| \le \sum_{i=1}^{n}\sum_{j=1}^{n} l_j|T_{ij}||u_j-v_j| \le \sum_{i=1}^{n} (\sum_{j=1}^{n} l_j|T_{ij}|) |u_i-v_i|$$

which gives

$$(k_1-k_2)\sum_{i=1}^{n} |u_i-v_i| \le 0 \cdot \tag{8}$$

It follows, from (8) that $u_i=v_i$ for $i=1,2,\cdots,n$, (i.e. $u=v$). Hence, the map $H \in C^0$ is injective on R^n.

Next, we prove that $\|H(u)\| \to \infty$ as $\|u\| \to \infty$, we consider the map $\hat{H}(u)=H(u)-H(0)$, i.e.

$$\hat{H}_i(u)=H_i(u)-H_i(0)=b_i(u_i)-\sum_{j=1}^{n} T_{ij} f_j(u_j),$$

for $u_i \in R$, $i=1,2,\cdots,n$. It is enough to show

$$\|\hat{H}(u)\| \to \infty \text{ as } \|u\| \to \infty.$$

We have

$$\sum_{i=1}^{n}\operatorname{sgn}(u_i)\hat{H}_i(u)=\sum_{i=1}^{n}\operatorname{sgn}(u_i)[b_i(u_i)-\sum_{j=1}^{n}T_{ij}f_j(u_j)]\le\sum_{i=1}^{n}[-\gamma_i|u_i|+\sum_{j=1}^{n}l_j|T_{ij}||u_j|]=-\sum_{i=1}^{n}[\gamma_i-\sum_{j=1}^{n}l_i|T_{ji}|]|u_i|\le-(k_1-k_2)\sum_{i=1}^{n}|u_i|,$$

that yields

$$(k_1-k_2)\sum_{i=1}^{n}|u_i|\le-\sum_{i=1}^{n}\operatorname{sgn}(u_i)\hat{H}_i(u)\le\sum_{i=1}^{n}\left|\hat{H}_i(u)\right|\cdot$$

It then follows that

$$\sum_{i=1}^{n}|u_i|\le\frac{1}{k_1-k_2}\sum_{i=1}^{n}\left|\hat{H}_i(u)\right|$$

and from which we assert $\left\|\hat{H}(u)\right\|\to\infty$ as $\|u\|\to\infty$. We conclude that the map $H\in C^0$ is a homeomorphism on R^n and this guarantees the existence of a unique solution $u^*\in R^n$ of the algebraic system (4) which defines the unique equilibrium state of the impulsive network (2). The proof is now complete.

Theorem 2. Let $T_{ij}\in R$, in addition to H1), H2), H3) and (5) hold, assume further that:

H5) $k_3=\min\limits_{1\le i\le n}\{\underline{a}_i\gamma_i\}>\frac{1}{2}\max\limits_{1\le j\le n}\{\sum\limits_{i=1}^{n}\overline{a}_i(l_j^p|T_{ij}|^q+l_j^{2-p}|T_{ij}|^{2-q}\}=k_4>0$.

H6) There exists a constant γ such that

$$\frac{\ln\gamma_k}{t_k-t_{k-1}}\le\gamma<\lambda,\ k=1,2,\cdots,$$

where $\gamma_k=\max\{1,|1+\gamma_{1k}|,\cdots,|1+\gamma_{nk}|\},k\in N$, the scalar $\lambda>0$ is the unique positive solution of the equation: $\lambda=k_3-k_4e^{\lambda t}$.

Then the equilibrium point u^* of system (2) is globally exponential stable.

Proof. We consider the following Lyapunov function

$$V(t)=\sum_{i=1}^{n}|y_i(t)|\ \cdot$$

We denote

$$\overline{V}(t)=\sup_{t-\tau\le s\le t}V(s)=\sup_{t-\tau\le s\le t}(\sum_{i=1}^{n}|y_i(s)|)$$

for $t\ge0$. The upper right Dini derivative $D^+V(t)$ of $V(t)$ along the solutions of system (6) is obtained as

$$
\begin{aligned}
&D^+V(t)\\
&=\sum_{i=1}^{n}(-a_i(y_i(t))[b_i'(\theta_i)|y_i(t)|-\sum_{j=1}^{n}T_{ij}(f_j(y_j(t-\tau_j(t))+u_j^*)-f_j(u_j^*)))]\\
&\le\sum_{i=1}^{n}(-a_i(y_i(t))[b_i'(\theta_i)|y_i(t)|-\sum_{j=1}^{n}|T_{ij}|l_j|y_j(t-\tau_j(t))|]\\
&\le-\sum_{i=1}^{n}\underline{a}_i\gamma_i|y_i(t)|+\sum_{i=1}^{n}\sum_{j=1}^{n}\overline{a}_i|T_{ij}|^{\frac{q}{2}}|T_{ij}|^{\frac{2-q}{2}}l_j^{\frac{p}{2}}l_j^{\frac{2-p}{2}}|y_j(t-\tau_j(t))|\\
&\le-\sum_{i=1}^{n}\underline{a}_i\gamma_i|y_i(t)|+\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\overline{a}_i[|T_{ij}|^q l_j{}^p+|T_{ij}|^{2-q}l_j{}^{2-p}]|y_j(t-\tau_j(t))|\\
&\le-\min_{1\le i\le n}\{\underline{a}_i\gamma_i\}\sum_{i=1}^{n}|y_i(t)|+\frac{1}{2}\max_{1\le j\le n}\{\sum_{j=1}^{n}\overline{a}_i[|T_{ij}|^q l_j{}^p+|T_{ij}|^{2-q}l_j{}^{2-p}]\}\sum_{i=1}^{n}|\overline{y}_i(t)|\\
&\le-k_3V(t)+k_4\overline{V}(t)
\end{aligned}
$$

According to assumption H6) and Lemma 3, it is obviously that

$$V(t)\le\overline{V}(t_0)e^{-\lambda(t-t_0)}=\overline{V}(0)e^{-\lambda t}$$

for $0=t_0<t<t_1$. Where λ is the unique positive solution of the equation $\lambda=k_3-k_4e^{\lambda t}$. Also, in view of condition (5), one has

$$u_i(t_1)-u_i^*=u_i(t_1^-)-u_i^*+I_{i1}(u_i(t_1^-)-u_i^*)=(1+\gamma_{i1})(u_i(t_1^-)-u_i^*)\,.$$

Then at $t=t_1$,

$$V(t_1^+)=\sum_{i=1}^{n}|y_i(t_1^+)|=\sum_{i=1}^{n}\gamma_{i1}|y_i(t_1^-)|\le\gamma_1\sum_{i=1}^{n}|y_i(t_1^-)|=\gamma_1V(t_1^-)\le\gamma_1V(0).$$

By following the similar inductive arguments as before, we derive that

$$V(t_{k-1}^+)\le\overline{V}(0)\,\gamma_0\gamma_1\gamma_2\cdots\gamma_{k-1}e^{-\lambda t_{k-1}}\,,$$

for $t=t_{k-1}, k\in N$, where $t_0=0$ and $\gamma_0=1$, by the mathematical induction, we can conclude that

$$V(t)\le\overline{V}(0)\,\gamma_0\gamma_1\gamma_2\cdots\gamma_{k-1}e^{-\lambda t}\,, \tag{9}$$

for $t_{k-1}<t<t_k, k\in N$. Noticing that $\gamma_k\le e^{\gamma(t_k-t_{k-1})}$ by H6), we can use (9) to conclude that $V(t)\le\overline{V}(0)e^{2\gamma(t_1-t_0)}e^{2\gamma(t_2-t_1)}\cdots e^{2\gamma(t_k-t_{k-1})}e^{-2\lambda t}\le\overline{V}(0)e^{2\gamma(t-t_0)}e^{-2\lambda t}=\overline{V}(0)\,e^{2\gamma t}e^{-2\lambda t}=\overline{V}(0)\,e^{-2(\lambda-\gamma)t}$, for $0<t<t_k, k\in N$, which is equivalent to

$$\sum_{i=1}^{n}|y_i(t)|\le\|\psi\|e^{-(\lambda-\gamma)t}\ \text{ for }\ 0<t<t_k, k\in N\ ,$$

where

$$\|\psi\|=\|\phi-u^*\|=\sup_{s\in[-\tau,0]}[\sum_{j=1}^{n}|\phi_j(s)-u_j^*|].$$

Hence

$$\sum_{i=1}^{n} |y(t)| \leq \|\psi\| e^{-(\lambda-\gamma)t}$$

for $0<t<t_k, k\in N$. So the vector solution $y(t)$ converges to the equilibrium state $y^*=0$ of the impulsive network (6) exponentially as $t\to\infty$, that is, the equilibrium point u^* of system (2) is globally exponentially stable. This completes the proof.

4 Conclusion

In this paper, we consider the exponential stability of a class of Cohen-Grossberg neural networks with time-varying delays and impulses. Under assumption that the activation function is globally Lipschitz continuous, we obtain some sufficient conditions for the Existence and globally exponential stability of a unique equilibrium.

References

1. Jiang, L.: Stability of Cohen-Grossberg Neural Networks with Time-variable Delay. Chinese Journal of Engineering Mathematics 26(2), 243–250 (2009)
2. Cao, J.D., Wang, J.: Global exponential stability and periodicity of recurrent neural networks with time delay. IEEE Trans. Circuits and Systems 152, 920–931 (2005)
3. Forti, M., Tesi, A.: New conditions for global stability of neural networks with application to linear and quadratic programing problems. IEEE Trans. Circuits Syst. I: Fund. Theor. Appl. 42, 354–366 (1995)
4. Hardy, G.H., Littlewood, J.E., Polya, G.: Inequalities. Cambridge University Press, London (1952)
5. Mohamad, S., Gopalsamy, K., Akca, H.: Exponential stability of artificial neural networks with distributed delays and large impulses. Nonlinear Analysis: Real World Applications 9, 872–888 (2008)
6. Yang, T.: Impulsive systems and control: Theory and Applications. Nova science publishers, Huntington (2001)
7. Akca, H., Alassar, R., Covacheva, V., Ai-Zahrani, E.: Continuous-time additive Hopfield-type neural networks with impulses. J. Math. Anal. Appl. 290, 436–451 (2004)
8. Gopalsamy, K.: Stability of artificial neural networks with impulses. Appl. Math. & Comp. 154, 783–813 (2004)
9. Xia, Y.H., Cao, J.D., Cheng, S.: Global exponential stability of delayed cellular neural networks with impulses. Neurocomputing 70, 2495–2501 (2007)
10. Yang, Z.C., Xu, D.Y.: Impulsive effects on stability of Cohen-Grossberg neural networks with variable delays. Applied Math. And Comput. 160, 1–16 (2005)

P-Moment Asymptotic Behavior of Nonautonomous Stochastic Differential Equation with Delay

Bing Li[1], Yafei Zhou[2], and Qiankun Song[1]

[1] Department of Mathematics, Chongqing Jiaotong University, Chongqing 400074, China
qiankunsong@163.com
[2] College of Mathematic and Software Science, Sichuan Normal University, Chengdu 610066, China

Abstract. In this paper, we consider p-moment asymptotic behaviors of a nonautonomous delay stochastic differential equation. By using L-operator differential inequality techniques, we get some sufficient criterions for p-moment ultimately bounded and exponential stability. These results are fit for stochastic neural networks model.

Keywords: Ultimate boundedness, L-operator, Exponential stability, Neural networks.

1 Introduction

In recent years, the ultimate boundedness for solutions and global attracting set of dynamical system has attracted more and more attention. It plays an important role in networks design, chaos, synchronization, system norm estimation, robust control and so on. A lot of relative results about functional differential equations could be searched in [1]- [8].

In fact, most processes may be effected by some random perturbation. We describe these perturbation in mathematics by "white noise" [13]. Subject to the time delay and stochastic perturbation, it is more exactly to study the stochastic differential equations with delay. According to the best of my knowledge, there is few results about ultimate boundedness and global attracting sets of stochastic differential equations with delay. Some general research such as stochastic ultimate boundedness come from [9].

In this paper, we try to establish some interesting methods for p-moments ultimate boundedness and global attracting sets of stochastic differential equations with delay as functional differential equations. In section (3), combining L-operator, we construct the inequality about the expect of the stochastic solution process. Using the inequality techniques([7] [10] [11]) and the properties of M-matrix([12] [13]), we obtain successfully some sufficient criterions for p-moments ultimate boundedness and global attracting sets of nonautonomous stochastic differential equations with delay. By our methods, we can count the boundary

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 561–568, 2010.

for p-moments of solutions of nonautonomous stochastic differential equations with delay. Especially, we can use our methods to deal with the p-moments exponential stability of nonautonomous stochastic differential equations with delay when the attracting set is zero.

2 Preliminaries and Model

Throughout this paper, R^n denotes the n-dimensional Euclidean space, $R_n^+ = [0,+\infty)\times\cdots\times[0,+\infty)$ and $C[X,Y]$ is the class of continuous mapping from the topological space X to the topological space Y.

In this paper, we consider the nonautonomous delay differential equation without stochastic perturbation as follows:

$$\begin{cases} dx(t) = [-A(t)x(t) + f(t,x,x_t)]dt, & t \geq 0 \\ x_0(s) = \phi(s), \quad -\tau \leq s \leq 0, \end{cases} \tag{1}$$

where $x(t) = col\{x_1,\cdots,x_n\}$, $f(t,x(t),x_t(s)) = col\{f_1(t,x,x_t),\cdots,f_n(t,x,x_t)\} \in C[R^+\times R^n\times R^n, R^n]$, $x_t \triangleq x(t+s)$, $-\tau \leq s \leq 0$. That is said $x_0(s) = x(s) = \phi(s)$, $-\tau \leq s \leq 0$ and τ is a positive number.

Clearly, the system (1) is a basic frame of dynamical system. There exist many results about the ultimate boundedness and global attracting set about this system([1]- [8]). But for describing the dynamic system more exactly, we must consider the stochastic perturbation to this function differential equations. Unless otherwise specified, we let $(\Omega, F, \{F_t\}_{t\geq 0}, P)$ be a complete probability space with a filtration $\{F_t\}_{t\geq 0}$ satisfying the usual conditions(i.e. it is right continuous and increasing while F_0 contains all P-null sets) and $C^\tau_{F_0}[[-\tau,0],R^n]$ be the family of all F_0 measurable bounded $C[[-\tau,0],R^n]$-valued random variables $\phi = \{\phi(s)|\ -\tau \leq s < 0\}$. We denote $C^\tau_{F_0} \triangleq C^\tau_{F_0}[[-\tau,0],R^n]$ if no confusion occurs. Let the n-dimensions Brownian motion defined on the space $(\Omega, F, \{F_t\}_{t\geq 0}, P)$ stochastically perturb the system (1). Consequently, we get the general nonautonomous stochastic differential equations with delay which describe motions exactly as follows:

$$\begin{cases} dx(t) = [-A(t)x(t) + f(t,x,x_t)]dt + \sigma(t,x,x_t(s))dw(t), & t \geq 0 \\ x_0(s) = \phi(s), \quad -\tau \leq s \leq 0, \end{cases} \tag{2}$$

in which $w(t) = col\{w_1(t),\cdots,w_n(t)\}$ is n-dimensions Brownian motion defined on the complete probability space with a filtration $\{F_t\}_{t\geq 0}$. The initial function $\phi(s) \in C^\tau_{F_0}$. The weight of stochastic perturbation $\sigma(t,\cdot) = (\sigma_{ij}(t,\cdot))_{n\times n}$, $\sigma_{ij}(t,\cdot) \in C[R^+\times R^n, R^n]$ is locally Lipschitz continuous and satisfies the linear growth condition([9]). So it is known that for any initial value $\phi \in C^\tau_{F_0}$, system (2) has an unique global solution on $t \geq 0$, which is denoted by $x(t,\phi)$ in this paper. Obviously, the solution $x(t,\phi)$ is a stochastic process respect to the probability space $(\Omega, F, \{F_t\}_{t\geq 0}, P)$.

For convenience, we denote $[x]^+ = col(|x_1|,\cdots,|x_n|)$, $[x]^p = col(|x_1|^p,\cdots,|x_n|^p)$. For any stochastic process $x(t)$, we denote $E[x]^p =$

$col(E|x_1|^p, \cdots, E|x_n|^p)$, in which $E(\cdot)$ is the expectation operator of stochastic process with respect to the probability space $(\Omega, F, \{F_t\}_{t\geq 0}, P)$. For each $\phi(s) = col\{\phi_1(s), \cdots, \phi_n(s)\} \in C^{\tau}_{F_0}$, $E[\phi(s)]^p_{\tau} = col\{E\|\phi_1(s)\|^p_{\tau}, \cdots, E\|\phi_n(s)\|^p_{\tau}\}$, where $E\|\phi_i(s)\|^p_{\tau} = \sup\limits_{-\tau\leq s\leq 0} E|\phi_i(s)|^p$.

According to the Ito formula, we defined that $V(t,x) = col(V_1(t,x), \cdots, V_n(t,x)) \in C[R^+ \times R^n, R^n_+]$, where $x(t)$ is a n-dimensions stochastic process. For each i, the $V_i(t,x) \in C^{2,1}[R^+ \times R^n, R^n_+]$ which denotes the family of all nonnegative functions on $R^+ \times R^n$ which are continuous twice differentiable in x and once differentiable in t. For each $V_i \in C^{2,1}$, we define an operator LV_i associated with the system (2) by

$$LV_i = (V_i)_t + (V_i)_x f(t,x,x_t) + \frac{1}{2} trace[\sigma^T (V_i)_{xx} \sigma]$$

where $(V_i)_t = \frac{\partial V_i}{\partial t}$, $(V_i)_x = (\frac{\partial V_i}{\partial x_1}, \cdots, \frac{\partial V_i}{\partial x_n})$, $(V_i)_{xx} = (\frac{\partial (V_i)}{\partial x_i x_j})_{n\times n}$.

Furthermore, for any $V(t,x) \in C[R^+ \times R^n, R^n_+]$, we denote $LV(t,x) = (LV_1(t,x), \cdots, LV_n(t,x))$. The inequality " $\leq$ "(" $>$ ") between matrices or vectors such as $A \leq B (A > B)$ means that each pair of corresponding elements of A and B satisfies the inequality " $\leq$ "(" $>$ "). Especially, A is called a nonnegative matrix(or vector) if $A \geq 0$.

In the following, we will give some definitions as those for the differential equation with delays given in [6] [7] [10] [11]:

Definition 1. *The system (2) is called p-moments ultimate boundedness if, there exists a constant vector $K > 0$, such that for any initial value $\phi \in C^{\tau}_{F_0}$, there is a $T(0,\phi)$, when $t > T(0,\phi)$, the solution $x(t,\phi)$ of system (2) satisfies*

$$E[x(t,\phi)]^p \leq K.$$

or

$$\limsup_{t\longrightarrow\infty} E[x(t,\phi)]^p = K$$

In this case, the set $\Omega = \{\phi \in C^{\tau}_{F_0} | E[\phi(s)]^p_{\tau} \leq K\}$ is said to be the p-moment global attracting set of (2).

Definition 2. *The solution of system (2) is p-moment exponential stable if, there exist a positive real number r and a constant vector $D > 0$, such that*

$$E[x(t)]^p \leq De^{-rt}, \quad t > 0.$$

Meanwhile, r is called the exponential convergence rate of system (2).

3 Asymptotic Behavior Analysis

In this section, combining the inequality technique in Lemma as follows with properties of M-matrices, we use the Ito formula and L operator to establish some sufficient criterions for the p-moment ultimately boundedness and exponential stability of system (2).

Lemma 1. *([11]) If $a_i \geq 0$, $b_i \geq 0$, $p > 0$, $q > 0$ and $\frac{1}{p} + \frac{1}{q} = 1$ then*

$$\sum_{i=1}^{n} a_i b_i \leq (\sum_{i=1}^{n} a_i^p)^{\frac{1}{p}} (\sum_{i=1}^{n} b_i^q)^{\frac{1}{q}}$$

Lemma 2. *([11]) For $x_i \geq 0$, $\alpha_i > 0$ and $\sum_{i=1}^{n} \alpha_i = 1$, we have*

$$\Pi_{i=1}^{n} x_i^{\alpha_i} \leq \sum_{i=1}^{n} \alpha_i x_i$$

Lemma 3. *([13]) If A is a M-matrix, then for positive vector $I > 0$, $A^{-1}I > 0$.*

Lemma 4. *Let $P = (p_{ij})_{n\times n}$ and $p_{ij} \geq 0$ when $i \neq j$, $Q = (q_{ij})_{n\times n} \geq 0$ are real matrix, $R = col(R_1, \cdots, R_n) \geq 0$. The vector function $u(t) = col(u_1(t), \cdots, u_n(t))$ satisfies the differential inequality following:*

$$Du(t) \leq h(t)[Pu(t) + Qu(t-\tau) + R], \qquad t \geq 0. \tag{3}$$

Where $D(\cdot)$ denotes the Dini derivation. $h(t)$ is a positive integral function and $\lim_{t\to\infty} \int_0^t h(s)ds = +\infty$. $\int_{t-\tau}^{t} h(s)ds$ is bounded for $\forall t > 0$. If $M = -(P + Q)$ is a M-matrix, then we have

$$u(t) \leq Ke^{-r\int_0^t h(s)ds} - (P+Q)^{-1}R, \quad t \geq 0. \tag{4}$$

provided that the initial conditions satisfies

$$u(t) \leq Ke^{-r\int_0^t h(s)ds} - (P+Q)^{-1}R, \quad -\tau \leq t \leq 0$$

where $K = col\{k_1, \cdots, k_n\}$ is a positive constant vector and r is determined by the inequality $(rE + P + Qe^{-r\int_{t-\tau}^{t} h(s)ds})K < 0$, E is an unit matrix.

Proof. The proof is similar as [8,10].

Theorem 1. *Let $P = (p_{ij})_{n\times n}$, $p_{ij} \geq 0$ for $i \neq j$. $Q = (q_{ij})_{n\times n} \geq 0$. $R = col(R_1, \cdots, R_n) \geq 0$. The positive integral function $h(t)$ satisfies $\lim_{t\to\infty} \int_{t_0}^{t} h(s)ds = +\infty$ and $\forall t > 0$, $\int_{t-\tau}^{t} h(s)ds$ is bounded. Assume there exists a vector function $V(t,x) = col(V_1(t,x), \cdots, V_n(t,x)) \in C^2[R^+ \times R^n, R_+^n]$, where $x(t)$ is a solution of system (2), such that*

$$LV_i(t,x) \leq h(t)[\sum_{j=1}^{n}(P_{ij}V_j(x) + Q_{ij}V_j(x_t)) + R_i]$$

If $-(P + Q)$ is a M-matrix, then we have

$$EV(t,x) \leq Ke^{-\lambda\int_0^t h(s)ds} - (P+Q)^{-1}R, \quad t > 0$$

provided that the initial function satisfies

$$EV(t,x) \leq Ke^{-\lambda\int_0^t h(s)ds} - (P+Q)^{-1}R, \quad -\tau \leq t \leq 0$$

Proof. Since $x(t)$ is a solution of system (2) and $V(t,x) \in C^2[R^+ \times R^n, R_+^n]$, by the Ito formula, we can get for $\forall i \in N$

$$V_i(t) = V_i(0) + \int_0^t LV_i(x(s))ds + \int_0^t \frac{\partial V_i(x(s))}{\partial x}\sigma(s, x(s), x_s)dw(t), \quad t > 0 \tag{5}$$

Counting expectation both sides of (5), we have

$$EV_i(t) = EV_i(0) + E\int_0^t LV_i(x(s))ds, \quad t > 0 \tag{6}$$

Let $\triangle t > 0$ is small enough, then we get

$$EV_i(t+\triangle t) - EV_i(t) = E\int_t^{t+\triangle t} LV_i(x(s))ds, \quad t > 0 \tag{7}$$

Due to Fubini Theorem, we know that

$$\begin{aligned} EV_i(t+\triangle t) - EV_i(t) &= \int_t^{t+\triangle t} ELV_i(x(s))ds \\ \leq \int_t^{t+\triangle t} h(s)[\sum_{j=1}^n (P_{ij}EV_j(x) + Q_{ij}EV_j(x_s)) + R_i]ds \end{aligned} \tag{8}$$

We can derived that

$$DEV_i(t) \leq h(t)[\sum_{j=1}^n (P_{ij}EV_j(x) + Q_{ij}EV_j(x_t) + R_i], \quad t > 0 \tag{9}$$

From Lemma 4, we complete the proof.

Next, we give some assumptions about system (2) for studying P-moment asymptotic behavior by using theorem above.

(A1). For $A(t)$, assume that $a_{ii}(t) \leq a_{ii}h(t)$, where $a_{ii} < 0$, $|a_{ij}(t)| \leq a_{ij}h(t)$, where $a_{ij} \geq 0$; $[f(t,x,x_t)]^+ \leq h(t)[B[x_t]^+ + I]$, where $B = (b_{ij})_{n\times n} \geq 0$, $I = col\{I_1, \cdots, I_n\} \geq 0$.
(A2). There are two nonnegative matrix $C = (c_{ij})_{n\times n}$ and $D = (d_{ij})_{n\times n}$ such that $|\sigma_i\sigma_i^T| \leq h(t)[\sum_{j=1}^n c_{ij}x_j^2(t) + \sum_{j=1}^n d_{ij}x_j^2(t-\tau)]$, where $\sigma_i = (\sigma_{1i}, \cdots, \sigma_{ni})$, $i = 1, \cdots, m$.
(A3). Positive function $h(t)$ satisfies $\lim\limits_{t\to\infty} \int_{t_0}^t h(s)ds = +\infty$ and $\forall t > 0$, $\int_{t-\tau}^t h(s)ds$ is bounded.
(A4). The matrix $-(P+Q)$ is a M-matrix, in which $P = (p_{ij})_{n\times n}$, $Q = (q_{ij})_{n\times n}$, $p_{ii} = pa_{ii} + (p-1)(\sum\limits_{i\neq j} a_{ij} + \sum_{j=1}^n b_{ij}) + p - 1 + (p-1)c_{ii} + \frac{1}{2}(p-1)(p-2)\sum_{j=1}^n (c_{ij} + d_{ij})$
$p_{ij} = a_{ij} + (p-1)c_{ij}, i \neq j; \quad q_{ij} = b_{ij} + (p-1)d_{ij}, \forall i, j \in N; \quad R = (I_1^p, \cdots, I_n^p)$.

Theorem 2. *Let assumptions(A1)-(A4) hold, then the solution process $x(t)$ of system (2) satisfies estimate as follows*

$$E[x(t)]^p \leq Ke^{-\lambda \int_0^t h(s)ds} - (P+Q)^{-1}R, \quad t > 0. \tag{10}$$

if the initial condition satisfies

$$E[x(t)]^p \leq Ke^{-\lambda \int_0^t h(s)ds} - (P+Q)^{-1}R, \quad -\tau \leq t \leq 0.$$

where λ is determined by $(rE + P + Qe^{r\int_{t-\tau}^t h(s)ds})z < 0$.

Proof. We choose $V_i(x) = |x_i(t)|^p (p \geq 2,\ i = 1, \cdots, n)$, where $x_i(t)$ is a solution of system (2). Obviously, we get

$$\frac{\partial V_i(x)}{\partial x_i} = p|x_i|^{p-1} sgn(x_i), \quad \frac{\partial^2 V_i(x)}{\partial x_i^2} = p(p-1)|x_i|^{p-2} sgn(x_i)$$

where $sgn(x_i)$ is the sign function.

Using assumptions (A1)(A2), we can obtain that

$$\begin{aligned} LV_i(x) &= p|x_i|^{p-2}x_i[\sum_{j=1}^n a_{ij}(t)x_j + f_i(t,x)] + \frac{1}{2}p(p-1)|x_i|^{p-2}sgn(x_i)\sigma_i\sigma_i^T \\ &\leq pa_{ii}(t)|x_i|^p + p|x_i|^{p-1}\sum_{i\neq j}|a_{ij}(t)||x_j| + p|x_i|^{p-1}h(t)\sum_{j=1}^n b_{ij}|x_j|_t \\ &\quad + p|x_i|^{p-1}I_i h(t) + \frac{1}{2}p(p-1)|x_i|^{p-2}h(t)[\sum_{j=1}^n c_{ij}x_j^2 + \sum_{j=1}^n d_{ij}|x_j|_t^2] \\ &\leq pa_{ii}h(t)|x_i|^p + p|x_i|^{p-1}h(t)\sum_{i\neq j}a_{ij}|x_j| + p|x_i|^{p-1}h(t)\sum_{j=1}^n b_{ij}|x_j|_t \\ &\quad + p|x_i|^{p-1}h(t)I_i + \frac{1}{2}p(p-1)|x_i|^{p-2}h(t)[\sum_{j=1}^n c_{ij}x_j^2 + \sum_{j=1}^n d_{ij}|x_j|_t^2] \end{aligned} \tag{11}$$

According to Lemma 1 and Lemma 2, we know that

$$\begin{aligned} LV_i(x) &\leq pa_{ii}h(t)|x_i|^p + p|x_i|^{p-1}h(t)\sum_{i\neq j}a_{ij}|x_j| + p|x_i|^{p-1}h(t)\sum_{j=1}^n b_{ij}|x_j|_t \\ &\quad + p|x_i|^{p-1}h(t)I_i + \frac{1}{2}p(p-1)|x_i|^{p-2}h(t)[\sum_{j=1}^n c_{ij}x_j^2 + \sum_{j=1}^n d_{ij}|x_j|_t^2] \\ &\leq h(t)pa_{ii}|x_i|^p + h(t)\sum_{i\neq j}a_{ij}[(p-1)|x_i|^p + |x_j|^p] \end{aligned}$$

$$+h(t)\sum_{j=1}^{n} b_{ij}[(p-1)|x_i|^p + |x_j|_t^p] + (p-1)|x_i|^p + I_i^p$$
$$+\frac{1}{2}(p-1)h(t)\sum_{j=1}^{n} c_{ij}[(p-2)|x_i|^p + 2|x_j|^p]$$
$$+\frac{1}{2}(p-1)h(t)\sum_{j=1}^{n} d_{ij}[(p-2)|x_i|^p + 2|x_j|_t^p] \tag{12}$$

It can be deduced from inequality above

$$\begin{aligned} LV_i(x) \le h(t)\{&[pa_{ii} + (p-1)(\sum_{i\neq j}|a_{ij}| + \sum_{j=1}^{n} b_{ij}) + p - 1 \\ &+(p-1)c_{ii} + \frac{1}{2}(p-1)(p-2)\sum_{j=1}^{n}(c_{ij}+d_{ij})]|x_i|^p \\ &+\sum_{i\neq j}[|a_{ij}| + (p-1)c_{ij}]|x_j|^p + \sum_{j=1}^{n}[|b_{ij}| + (p-1)d_{ij}]|x_j|_\tau^p + I_i^p\} \\ = h(t)&[\sum_{j=1}^{n}(p_{ij}V_j(x) + q_{ij}V_j(x_t)) + R_i] \end{aligned} \tag{13}$$

From theorem (1), we can obtain that

$$E[x(t)]^p \le Ke^{-\lambda\int_0^t h(s)ds} - (P+Q)^{-1}R, \quad t > 0.$$

The proof is completed.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 60974132 and 10671133 , Youth Foundation of Chongqing Jiaotong University (2008L-02) and Education Improvement Foundation of Chongqing Jiaotong University (0802001).

References

1. Arik, S.: On The Global Dissipativity of Dynamical Neural Networks with Time Delays. Phys. Lett. A. 326, 126–132 (2004)
2. Cao, J.D., Yuan, K., Ho, D.W.C., Lam, J.: Global Point Dissipativity of Neural Networks with Mixed Time-varying Delays. Chaos 16, 013105 (2006)
3. Xu, D.Y., Zhao, H.Y.: Invariant and Attracing Sets of Hopfield Neural Networks with Delay. Int. J. Syst. Sci. 32, 863–866 (2001)
4. Xu, D.Y.: Integro-differential Equations and Delay Integral Inequalities. Tohoku Math. J. 44, 365–378 (1992)

5. Xu, D.Y.: Asymptotic Behavior of Hopfield Neural Networks with Delays. Differential Equations Dyn. Syst. 9, 353–364 (2001)
6. Liao, X.X., Wang, J.: Global Dissipativity of Continuous-time Recurrent Neural Networks with Time Delays. Phys. Rev. E. 68, 1–7 (2003)
7. Huang, Y.M., Xu, D.Y., Yang, Z.C.: Dissipativity and Periodic Attractor for Non-autonomous Neural Networks with Time-varying Delays. Neurocomputing 70, 2953–2958 (2007)
8. Li, B., Xu, D.Y.: Dissipativity of Neural Networks with Continuously Distributed Delays. Electron. J. Differential Equations 119, 1–7 (2006)
9. Mao, X.R.: Attraction, Stability and Boundedness for Stochastic Differential Delay Equations. Nonlinear Anylysis 47, 4795–4806 (2001)
10. Li, B., Xu, D.Y.: Mean Square Asymptotic Behavior of Stochastic Neural Networks with Infinitely Distributed Delays. Neurocomputing 72, 3311–3317 (2009)
11. Xu, L.G., Xu, D.Y.: P-attracting and P-invariant Sets for A Class of Impulsive Stochastic Functional Differential Equations. Computers and Mathematics with Applications 57, 54–61 (2009)
12. Lasalle, J.P.: The Stability of Dynamical System. SIAM, Philadelphia (1976)
13. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1985)

Exponential Stability of the Neural Networks with Discrete and Distributed Time-Varying Delays

Qingbo Li[1], Peixu Xing[1], and Yuanyuan Wu[2]

[1] Department of Mathematics and Information Science,
Zhengzhou University of Light Industry, Zhengzhou 450002, China
[2] School of Electrical and Electronic Engineering,
Zhengzhou University of Light Industry, Zhengzhou 450002, China

Abstract. For a class of generalized neural networks(NNs) with discrete and distributed time-varying delays, this paper is concerned with the problem of the global exponential stability analysis. By introducing a novel augmented Lyapunov-Krasovskii functional and some appropriate free-weighting matrices, a new delay-dependent stability criterion is derived in terms of linear matrix inequalities (LMIs). Finally, a numerical example is given to show the superiority of the obtained results.

Keywords: Neural networks, Exponential stability, Delay-dependent, Free-weighting matrix, LMIs.

1 Introduction

Recently, there has been a rapidly growing research interest on neural networks (NNs), for it has successful applications in many areas, especially in various signal processing problems. It is well-known that time delay is a inherent feature of neural networks and its existence often leads to instability, oscillation, and poor performances of neural networks. Hence, increasing attention has been focused on stability problem of neural networks with time delays[1,2,3,4,5,6,7,8,9,10,11,12].

Considering practical applications, when modeling neural networks, both the discrete and distributed time delays should be taken into account[3]. Therefore, much attention has been paid to the kind of NNs[3,4,5,6,7,8,9]. For example, Liu et al. [4] has considered the global exponential stability analysis of generalized recurrent neural networks. However, the delays considered in [2,4,7,12] are all constants. We know, in most situations, the studies of NNs with time-varying delays are more realistic than those with constant ones.

In this paper, the main purpose is to study the global exponential stability analysis of generalized neural networks with discrete and distributed time-varying delays. A delay-dependent criterion is derived for the exponential stability, meanwhile, the corresponding convergence rate can be estimated. Finally, resorting to LMI in the Matlab toolbox, the effectiveness and improvement of our results can be checked by numerical examples.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 569–576, 2010.

Throughout this paper, $\|\cdot\|$ denotes the Euclidean norm. A real symmetric matrix $P > 0 (\geq 0)$ denotes P being a positive definite (positive semi-definite) matrix, $\lambda_{max}(P)$ and $\lambda_{min}(P)$ represent for the maximum and minimum eigenvalues of the matrix P respectively. The symmetric terms in a symmetric matrix are denoted by $*$.

2 Problem Formulations

Consider the following neural networks with discrete and distributed delays

$$\dot{x}(t) = -Cx(t) + Af_1(x(t)) + Bf_2(x(t - d(t))) + D\int_{t-\tau(t)}^{t} f_3(x(s))ds + u, \quad (1)$$

where $x(\cdot) = [x_1(\cdot),\ x_2(\cdot), \cdots, x_n(\cdot)]^T \in R^n$ is the neuron state vector, $f_i(x(\cdot)) = [f_{i1}(x_1(\cdot)),\ f_{i2}(x_2(\cdot)), \cdots, f_{in}(x_n(\cdot))]^T \in R^n\ (i = 1, 2, 3)$ denote the neuron activation functions, $u = [u_1,\ u_2, \cdots,\ u_n]^T \in R^n$ is a constant input vector. $C = diag\{c_1,\ c_2, \cdots,\ c_n\}$ is a positive diagonal matrix, and A, B, D denote, respectively, the connection weight matrix, the discretely delayed connection weight matrix and the distributively delayed connection weight matrix, $d(t)$, $\tau(t)$ are the time-varying delays, and satisfying $0 \leq d(t) \leq d_M,\ \dot{d}(t) \leq \mu,\ \tau(t) \leq \tau_M$, where d_M, μ, and τ_M are constants.

For $i \in \{1, 2, ..., n\},\ \forall\ x, y \in R, x \neq y$, the neutron activation functions in (1) are assumed to be bounded, and satisfy

$$l_i^- \leq \frac{f_{1i}(x)-f_{1i}(y)}{x-y} \leq l_i^+,\ \sigma_i^- \leq \frac{f_{2i}(x)-f_{2i}(y)}{x-y} \leq \sigma_i^+,\ \upsilon_i^- \leq \frac{f_{3i}(x)-f_{3i}(y)}{x-y} \leq \upsilon_i^+,$$

where $l_i^-, l_i^+, \sigma_i^-, \sigma_i^+, \upsilon_i^-, \upsilon_i^+$ are some constants.

Remark 1. In this paper, it is assumed that the derivative of time-varying delay $d(t)$ can take any value, and the above constants $l_i^-, l_i^+, \sigma_i^-, \sigma_i^+, \upsilon_i^-, \upsilon_i^+$ are allowed to be positive, negative or zero, which make the paper have better applicability.

Obviously, system (1) has one equilibrium point[5]. Assume $x^* = [x_1^*,\ x_2^*, \cdots, x_n^*]^T \in R^n$ is the equilibrium point of system (1). We make the transformation $z(\cdot) = x(\cdot) - x^*$, then system (1) can be transformed into

$$\dot{z}(t) = -Cz(t) + Ag_1(z(t)) + Bg_2(z(t - d(t))) + D\int_{t-\tau(t)}^{t} g_3(z(s))ds, \quad (2)$$

where $z(\cdot) = [z_1(\cdot),\ z_2(\cdot), \cdots, z_n(\cdot)]^T$ is the state vector of the transformed system (2), and the transformed neuron activation functions are $g_i(z(\cdot)) = [g_{i1}(z_1(\cdot)),\ g_{i2}(z_2(\cdot)), \cdots, g_{in}(z_n(\cdot))]^T = f_i(z(\cdot) + x^*) - f_i(x^*)(i = 1, 2, 3)$. According to assumptions above, it is easy to derive that

$$l_i^- \leq \frac{g_{1i}(x)-g_{1i}(y)}{x-y} \leq l_i^+,\ \sigma_i^- \leq \frac{g_{2i}(x)-g_{2i}(y)}{x-y} \leq \sigma_i^+,\ \upsilon_i^- \leq \frac{g_{3i}(x)-g_{3i}(y)}{x-y} \leq \upsilon_i^+.$$

Let the initial conditions be $z(s) = \phi(s),\ s \in [-\tau^*,\ 0],\ \tau^* = \max\{d_M, \tau_M\}$, where $\phi(s)$ is a continuous real-valued function on its domain. We are now to introduce the notion of the global exponential stability for the system (2).

Definition 1. *The equilibrium point x^* is said to be globally exponentially stable, if there exist scalars $k > 0$ and $r > 0$ such that*

$$\|z(t)\| \leq re^{-kt}\|\phi\|, \quad \forall t > 0, \tag{3}$$

where $\|\phi\| = \sup_{-\tau^* \leq s \leq 0} \|\phi(s)\|$, scalar $k > 0$ is called the exponential convergence rate.

3 Main Results

To establish the main results of the paper, the following lemma will be used.

Lemma 1. *For any constant matrix $W \in R^{m\times m}, W = W^T > 0$, scalar $0 < r(t) < r$, vector function $\omega : [0, r] \to R^m$ such that the integrations concerned are well defined, then*

$$r(t)\int_0^{r(t)} \omega^T(s)W\omega(s)ds \geq (\int_0^{r(t)} \omega(s)ds)^T W(\int_0^{r(t)} \omega(s)ds). \tag{4}$$

For presentation convenience, in the following, we denote $\Lambda_1 = diag\{l_1^-, \cdots, l_n^-\}, \Lambda_2 = diag\{\sigma_1^-, \cdots, \sigma_n^-\}, \Lambda_3 = diag\{\upsilon_1^-, \cdots, \upsilon_n^-\}; \bar{\Lambda}_1 = diag\{l_1^+, \cdots, l_n^+\}, \bar{\Lambda}_2 = diag\{\sigma_1^+, \cdots, \sigma_n^+\}, \bar{\Lambda}_3 = diag\{\upsilon_1^+, \cdots, \upsilon_n^+\}; \tilde{l}_i = \max\{|l_i^-|, |l_i^+|\}, \tilde{\Lambda}_1 = diag\{\tilde{l}_1, \cdots, \tilde{l}_n\}, \tilde{\sigma}_i = \max\{|\sigma_i^-|, |\sigma_i^+|\}, \tilde{\Lambda}_2 = diag\{\tilde{\sigma}_1, \cdots, \tilde{\sigma}_n\}, \tilde{\upsilon}_i = \max\{|\upsilon_i^-|, |\upsilon_i^+|\}, \tilde{\Lambda}_3 = diag\{\tilde{\upsilon}_1, \cdots, \tilde{\upsilon}_n\}$, where $i \in \{1, 2, \cdots, n\}$, and

$$\begin{aligned}
&\Sigma_1 = diag\{l_1^+ l_1^-, \cdots, l_n^+ l_n^-\}, \quad \Sigma_2 = diag\{\frac{l_1^+ + l_1^-}{2}, \cdots, \frac{l_n^+ + l_n^-}{2}\},\\
&\Sigma_3 = diag\{\sigma_1^+ \sigma_1^-, \cdots, \sigma_n^+ \sigma_n^-\}, \quad \Sigma_4 = diag\{\frac{\sigma_1^+ + \sigma_1^-}{2}, \cdots, \frac{\sigma_n^+ + \sigma_n^-}{2}\},\\
&\Sigma_5 = diag\{\upsilon_1^+ \upsilon_1^-, \cdots, \upsilon_n^+ \upsilon_n^-\}, \quad \Sigma_6 = diag\{\frac{\upsilon_1^+ + \upsilon_1^-}{2}, \cdots, \frac{\upsilon_n^+ + \upsilon_n^-}{2}\}.
\end{aligned} \tag{5}$$

Theorem 1. *For given scalars $\mu > 0$, $d_M > 0$, and $\tau_M > 0$, the equilibrium point of the system (2) is globally exponentially stable, if there exist matrices $P > 0$, $Q_i > 0 (i = 1, 2, 3, 4)$, $G = \begin{bmatrix} G_{11} & G_{12}^T \\ G_{12} & G_{22} \end{bmatrix} > 0$, $H = \begin{bmatrix} H_{11} & H_{12}^T \\ H_{12} & H_{22} \end{bmatrix} > 0$, diagonal matrices $U_i > 0 (i = 1, 2, 3, 4)$, $R_i > 0 (i = 1, 2, 3)$, and appropriately dimensional matrices $M_j = \begin{bmatrix} M_{j1} \\ M_{j2} \end{bmatrix} (j = 1, 2)$, such that the following inequalities hold*

$$\begin{bmatrix} \Omega + \Xi + \Xi^T & d_M \Pi^T & d_M M_l^T \\ * & -d_M G_{22} & 0 \\ * & * & -d_M G \end{bmatrix} < 0, \tag{6}$$

where $l = 1$ or 2, and

$$\Omega = \begin{bmatrix} \Omega_{11} & 0 & 0 & H_{11}^T & H_{12}^T & \Omega_{16} & \Omega_{17} & \Omega_{18} & \Omega_{19} & \Omega_{1,10} \\ * & \Omega_{22} & 0 & \Omega_{24} & \Omega_{25} & 0 & 0 & U_4\Sigma_4 & 0 & 0 \\ * & * & -Q_2 & -H_{12} & -H_{22}^T & 0 & 0 & 0 & 0 & 0 \\ * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & * & \Omega_{66} & A^T R_2 & R_1 B & A^T R_3 & R_1 D \\ * & * & * & * & * & * & Q_3 - U_2 & R_2 B & 0 & R_2 D \\ * & * & * & * & * & * & * & \Omega_{88} & B^T R_3 & 0 \\ * & * & * & * & * & * & * & * & \tau_M^2 Q_4 - U_3 & R_3 D \\ * & * & * & * & * & * & * & * & * & -Q_4 \end{bmatrix} \tag{7}$$

with

$$\begin{aligned}
\Pi &= \begin{bmatrix} -G_{22}C & 0 & 0 & 0 & 0 & G_{22}A & 0 & G_{22}B & 0 & G_{22}D \end{bmatrix}, \\
\Xi &= \begin{bmatrix} M_{12}^T & -M_{12}^T + M_{22}^T & -M_{22}^T & M_{11}^T & M_{21}^T & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \\
\Omega_{11} &= -PC - C^T P + Q_1 + Q_2 + 2[R_1\Lambda_1 + R_2\Lambda_2 + R_3\Lambda_3]C - U_1\Sigma_1 - U_2\Sigma_3 \\
&\quad -U_3\Sigma_5 + d_M G_{11} - d_M G_{12}^T C - d_M C^T G_{12}, \\
\Omega_{16} &= PA - C^T R_1 - (\Lambda_1 R_1 + \Lambda_2 R_2 + \Lambda_3 R_3)A + U_1\Sigma_2 + d_M G_{12}^T A, \\
\Omega_{17} &= -C^T R_2 + U_2\Sigma_4, \quad \Omega_{18} = PB - (\Lambda_1 R_1 + \Lambda_2 R_2 + \Lambda_3 R_3)B + d_M G_{12}^T B, \\
\Omega_{19} &= -C^T R_3 + U_3\Sigma_6, \quad \Omega_{1,10} = PD - (\Lambda_1 R_1 + \Lambda_2 R_2 + \Lambda_3 R_3)D + d_M G_{12}^T D, \\
\Omega_{22} &= -(1-\mu)Q_1 - U_4\Sigma_3, \quad \Omega_{24} = -(1-\mu)H_{11}^T + (1-\mu)H_{12}, \\
\Omega_{25} &= -(1-\mu)H_{12}^T + (1-\mu)H_{22}, \quad \Omega_{66} - R_1 A + A^T R_1 - U_1, \\
\Omega_{88} &= -(1-\mu)Q_3 - U_4.
\end{aligned}$$

Proof. Firstly, we introduce the following Lyapunov-Krasovskii functional

$$V(t) = V_1(t) + V_2(t) + V_3(t) + V_4(t) + V_5(t) + V_6(t), \tag{8}$$

where

$$\begin{aligned}
V_1(t) &= z^T(t)Pz(t) + \alpha^T(t)H\alpha(t), \\
V_2(t) &= \int_{t-d(t)}^{t} z^T(s)Q_1 z(s)ds + \int_{t-d_M}^{t} z^T(s)Q_2 z(s)ds, \\
V_3(t) &= 2\sum_{i=1}^{n}\{r_{1i}\int_0^{z_i(t)}(g_{1i}(s) - l_i^- s)ds + r_{2i}\int_0^{z_i(t)}(g_{2i}(s) - \sigma_i^- s)ds \\
&\quad + r_{3i}\int_0^{z_i(t)}(g_{3i}(s) - \upsilon_i^- s)ds\}, \\
V_4(t) &= \int_{t-d(t)}^{t} g_2^T(z(s))Q_3 g_2(z(s))ds,
\end{aligned}$$

$$V_5(t) = \tau_M \int_{-\tau_M}^{0} \int_{t+\theta}^{t} g_3^T(z(s)) Q_4 g_3(z(s)) ds d\theta,$$
$$V_6(t) = \int_{-d_M}^{0} \int_{t+\theta}^{t} \beta^T(s) G \beta(s) ds d\theta,$$

with $\alpha^T(t) = [(\int_{t-d(t)}^{t} z(s)ds)^T, (\int_{t-d_M}^{t-d(t)} z(s)ds)^T]$, $\beta^T(s) = [z^T(s), \dot{z}^T(s)]$.

The time derivatives of $V_i(t)$ along the trajectory of system (2) yield

$$\dot{V}_1(t) = 2z^T(t)P[-Cz(t) + Ag_1(z(t)) + Bg_2(z(t-d(t)) + D\int_{t-\tau(t)}^{t} g_3(z(s))ds] + 2\alpha^T(t)H \begin{bmatrix} z(t) - (1-\mu)z(t-d(t)) \\ (1-\mu)z(t-d(t)) - z(t-d_M) \end{bmatrix}, \tag{9}$$

$$\dot{V}_2(t) \le z^T(t)(Q_1+Q_2)z(t) - (1-\mu)z^T(t-d(t))Q_1 z(t-d(t)) - z^T(t-d_M)Q_2 z(t-d_M), \tag{10}$$

$$\dot{V}_3(t) \le 2[g_1^T(z(t))R_1 + g_2^T(z(t))R_2 + g_3^T(z(t))R_3 - z^T(t)(\Lambda_1 R_1 + \Lambda_2 R_2 + \Lambda_3 R_3)][-Cz(t) + Ag_1(z(t)) + Bg_2(z(t-d(t)) + D\int_{t-\tau(t)}^{t} g_3(z(s))ds], \tag{11}$$

$$\dot{V}_4(t) \le g_2^T(z(t))Q_3 g_2(z(t)) - (1-\mu)g_2^T(z(t-d(t))Q_3 g_2(z(t-d(t)), \tag{12}$$

$$\dot{V}_5(t) \le \tau_M^2 g_3^T(z(t))Q_3 g_3(z(t)) - (\int_{t-\tau(t)}^{t} g_3(z(s))ds)^T Q_3 (\int_{t-\tau(t)}^{t} g_3(z(s))ds), \tag{13}$$

$$\dot{V}_6(t) \le d_M[z^T(t)G_{11}z(t) + 2z^T(t)G_{12}^T(-Cz(t) + Ag_1(z(t)) + Bg_2(z(t-d(t)) + D\int_{t-\tau(t)}^{t} g_3(z(s))ds) + \xi^T(t)\Pi^T G_{22}^{-1}\Pi\xi(t)] - \int_{t-d(t)}^{t} \beta^T(s)G\beta(s)ds - \int_{t-d_M}^{t-d(t)} \beta^T(s)G\beta(s)ds, \tag{14}$$

where $\Lambda_1, \Lambda_2, \Lambda_3$, and Π have been defined before, and $\xi^T(t) = [z^T(t),\ z^T(t-d(t)),\ z^T(t-d_M),\ (\int_{t-d(t)}^{t} z(s)ds)^T,\ (\int_{t-d_M}^{t-d(t)} z(s)ds)^T,\ g_1^T(z(t)),\ g_2^T(z(t)),\ g_2^T(z(t-d(t))),\ g_3^T(z(t)),\ (\int_{t-d(t)}^{t} g_3(s)ds)^T]$.

Moreover, the enlargement of two following integral terms can be conducted

$$\begin{aligned} &-\int_{t-d(t)}^{t} \beta^T(s)G\beta(s)ds \\ &\le 2\xi^T(t)M_{11}^T \int_{t-d(t)}^{t} z(s)ds + 2\xi^T(t)M_{12}^T[z(t) - z(t-d(t))] \\ &\quad + d(t)\xi^T(t)M_1^T G^{-1} M_1 \xi(t), \\ &\quad - \int_{t-d_M}^{t-d(t)} \beta^T(s)G\beta(s)ds \end{aligned} \tag{15}$$

$$\leq 2\xi^T(t)M_{21}^T\int_{t-d_M}^{t-d(t)} z(s)ds + 2\xi^T(t)M_{22}^T[z(t-d(t)) - z(t-d_M)]$$
$$+(d_M - d(t))\xi^T(t)M_2^T G^{-1}M_2\xi(t). \tag{16}$$

It can be seen that there exit diagonal matrices $U_i > 0 (i = 1, 2, 3, 4)$ such that the following inequalities hold(see [5,7,8] for detail).

$$\begin{bmatrix} z(t) \\ g_1(z(t)) \end{bmatrix}^T \begin{bmatrix} -U_1\Sigma_1 & U_1\Sigma_2 \\ * & -U_1 \end{bmatrix} \begin{bmatrix} z(t) \\ g_1(z(t)) \end{bmatrix} \geq 0, \tag{17}$$

$$\begin{bmatrix} z(t) \\ g_2(z(t)) \end{bmatrix}^T \begin{bmatrix} -U_2\Sigma_3 & U_2\Sigma_4 \\ * & -U_2 \end{bmatrix} \begin{bmatrix} z(t) \\ g_2(z(t)) \end{bmatrix} \geq 0, \tag{18}$$

$$\begin{bmatrix} z(t) \\ g_3(z(t)) \end{bmatrix}^T \begin{bmatrix} -U_3\Sigma_5 & U_3\Sigma_6 \\ * & -U_3 \end{bmatrix} \begin{bmatrix} z(t) \\ g_3(z(t)) \end{bmatrix} \geq 0, \tag{19}$$

$$\begin{bmatrix} z(t-d(t)) \\ g_2(z(t-d(t))) \end{bmatrix}^T \begin{bmatrix} -U_4\Sigma_3 & U_4\Sigma_4 \\ * & -U_4 \end{bmatrix} \begin{bmatrix} z(t-d(t)) \\ g_2(z(t-d(t))) \end{bmatrix} \geq 0, \tag{20}$$

where $\Sigma_i (i = 1, 2, 3, 4, 5, 6)$ are defined in (5).

Therefore, the time derivative of Lyapunov-Krasovskii functional $V(t)$ yields

$$\dot{V}(t) \leq \xi^T(t)\{\Omega + \Xi + \Xi^T + (d_M\Pi)^T(d_M G_{22})^{-1}(d_M\Pi) + d(t)M_1^T G^{-1}M_1$$
$$+(d_M - d(t))M_2^T G^{-1}M_2\}\xi(t) \tag{21}$$

where Ω, and Ξ are defined in Theorem 1. If (6) hold, then we have

$$\Delta_j = \Omega + \Xi + \Xi^T + (d_M\Pi)^T(d_M G_{22})^{-1}(d_M\Pi) + d_M M_j^T G^{-1}M_j < 0, \tag{22}$$

where $j = 1, 2$. Define $\varepsilon = \max\{\lambda_{max}(\Delta_j)\}$, then we know that $\dot{V}(t) \leq \epsilon\|\xi(t)\|^2 \leq \varepsilon\|z(t)\|^2$. Letting $\bar{V}(t) = e^{2kt}V(t)$, then

$$\bar{V}(t) \leq V(0) + \int_0^t [2ke^{2ks}V(s) + e^{2ks}\varepsilon\|z(s)\|^2]ds. \tag{23}$$

Let $\bar{\tau} = \max\{2d_M, d_M + \tau_M\}$, and it follows that

$$\dot{z}^T(s)\dot{z}(s) \leq 4[\lambda_{max}(C^T C)\|z(s)\|^2 + \lambda_{max}(A^T A)\lambda_{max}(\tilde{\Lambda}_1^T\tilde{\Lambda}_1)\|z(s)\|^2$$
$$+\lambda_{max}(B^T B)\lambda_{max}(\tilde{\Lambda}_2^T\tilde{\Lambda}_2)\|z(s-d(s))\|^2$$
$$+\tau_M\lambda_{max}(D^T D)\lambda_{max}(\tilde{\Lambda}_3^T\tilde{\Lambda}_3)\int_{s-\tau_M}^{s}\|z(\nu)\|^2 d\nu]. \tag{24}$$

We denote $\delta = \lambda_{max}(C^T C) + \lambda_{max}(A^T A)\lambda_{max}(\tilde{\Lambda}_1^T\tilde{\Lambda}_1) + \lambda_{max}(B^T B)$ $\times \lambda_{max}(\tilde{\Lambda}_2^T\tilde{\Lambda}_2) + \tau_M^2\lambda_{max}(D^T D)\lambda_{max}(\tilde{\Lambda}_3^T\tilde{\Lambda}_3)$, then

$$\int_{s-d_M}^{s}\beta^T(\nu)\beta(\nu)d\nu \leq \int_{s-\bar{\tau}}^{s}\|z(\nu)\|^2 d\nu + 4\delta\int_{s-\bar{\tau}}^{s}\|z(\nu)\|^2 d\nu. \tag{25}$$

Combined with (24) and (25), we can obtain

$$V(s) \leq \Theta_0 \|z(s)\|^2 + \Theta_1 \int_{s-\bar{\tau}}^{s} \|z(\nu)\|^2 d\nu, \tag{26}$$

where

$$\begin{aligned}\Theta_0 &= \lambda_{max}(P) + \lambda_{max}((\bar{\Lambda}_1 - \Lambda_1)R_1) + \lambda_{max}((\bar{\Lambda}_2 - \Lambda_2)R_2) \\ &\quad + \lambda_{max}((\bar{\Lambda}_3 - \Lambda_3)R_3), \\ \Theta_1 &= d_M \lambda_{max}(H) + \lambda_{max}(Q_1) + \lambda_{max}(Q_2) + \lambda_{max}(Q_3)\lambda_{max}(\tilde{\Lambda}_2^T \tilde{\Lambda}_2) \\ &\quad + \tau_M^2 \lambda_{max}(Q_4)\lambda_{max}(\tilde{\Lambda}_3^T \tilde{\Lambda}_3) + d_M(4\delta + 1)\lambda_{max}(G).\end{aligned}$$

Using the similar methods, $V(0)$ yields

$$V(0) \leq \vartheta \|\phi\|^2, \tag{27}$$

where

$$\begin{aligned}\vartheta &= \lambda_{max}(P) + d_M^2 \lambda_{max}(H) + d_M[\lambda_{max}(Q_1) + \lambda_{max}(Q_2) \\ &\quad + \lambda_{max}(Q_3)\lambda_{max}(\tilde{\Lambda}_2^T \tilde{\Lambda}_2) + d_M(4\delta + 1)\lambda_{max}(G)] + \lambda_{max}((\bar{\Lambda}_1 - \Lambda_1)R_1) \\ &\quad + \lambda_{max}((\bar{\Lambda}_2 - \Lambda_2)R_2) + \lambda_{max}((\bar{\Lambda}_3 - \Lambda_3)R_3) + \tau_M^3 \lambda_{max}(Q_4)\lambda_{max}(\tilde{\Lambda}_3^T \tilde{\Lambda}_3).\end{aligned}$$

Then by (23-27), we get

$$\bar{V}(t) \leq (\vartheta + 2k\Theta_1 \bar{\tau}^2 e^{2k\bar{\tau}})\|\phi\|^2 + (\varepsilon + 2k\Theta_0 + 2k\Theta_1 \bar{\tau} e^{2k\bar{\tau}}) \int_0^t e^{2ks} \|z(s)\|^2 ds. \tag{28}$$

Choose a constant k_0 satisfy $\varepsilon + 2k_0\Theta_0 + 2k_0\Theta_1 \bar{\tau} e^{2k_0\bar{\tau}} \leq 0$, then $\bar{V}(t) \leq (\vartheta + 2k_0\Theta_1 \bar{\tau}^2 e^{2k_0\bar{\tau}})\|\phi\|^2$. On the other hand, we have $\bar{V}(t) \geq e^{2k_0 t}\lambda_{min}(P)\|z(t)\|^2$, then it follows that $\|z(t)\| \leq \sqrt{\frac{\vartheta + 2k_0\Theta_1 \bar{\tau}^2 e^{2k_0\bar{\tau}}}{\lambda_{min}(P)}}\|\phi\| e^{-k_0 t}$. Then, by definition 1, the system (2) is exponentially stable with a convergence rate k_0, and our proof is completed.

4 Numerical Example

In the section, a numerical example will be given to illustrate the main results.

Example 1. Consider a third-order delayed neural networks(6)[9], where

$$C = \begin{bmatrix} 2.3 & 0 & 0 \\ 0 & 3.4 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}, \quad A = \begin{bmatrix} 0.9 & -1.5 & 0.1 \\ -1.2 & 1 & 0.2 \\ 0.2 & 0.3 & 0.8 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.8 & 0.6 & 0.2 \\ 0.5 & 0.7 & 0.1 \\ 0.2 & 0.1 & 0.5 \end{bmatrix}, \quad D = \begin{bmatrix} 0.3 & 0.2 & 0.1 \\ 0.1 & 0.2 & 0.1 \\ 0.1 & 0.1 & 0.2 \end{bmatrix},$$

$g_{1i}(s) = g_{2i}(s) = g_{3i}(s) = 0.1 \times (|x_i + 1| - |x_i - 1|), i = 1, 2, 3, d(t) = \tau(t).$

By applying Theorem 1 to this example, we can conclude that the system is global exponentially stable or not, and the maximum allowable bound d_M of delay can be obtained for several μ. [9] has concerned the asymptotic stability of the system. For different μ as 0, 0.5, 0.9, various maximum allowable bounds d_M of delay in [9] are 23.78, 22.67, 18.00, respectively. However, our corresponding results are 26.05, 25.12, 21.11. It is obvious that our criterion is less conservative than those in [9].

5 Conclusions

In the paper, we have investigated the global exponential stability problem of the generalized neural networks with discrete and distributed time-varying delays. By employing an augmented Lyapunov-Krasovskii functional, we proposed a novel stability criterion for the considered systems. Finally, a numerical example is given to show the superiority of proposed results.

References

1. Cao, J., Wang, J.: Global asymptotic and robust stability of recurrent neural networks with time delays. IEEE Trans. Circuits Syst. I 52, 417–426 (2005)
2. Chen, T., Rong, L.: Delay-independent stability analysis of Cohen- Grossberg neural networks. Phys. Lett. A 317, 436–449 (2003)
3. Cao, J., Yuan, K., Li, H.: Global asymptotical stability of recurrent neural networks with multiple discrete delays and ditributed delays. IEEE Trans. Neural Netw. 17, 1646–1651 (2006)
4. Liu, Y., Wang, Z., Liu, X.: Global exponential stability of generalized recurrent neural networks with discrete and distributed delays. Neural Networks 19, 667–675 (2006)
5. Li, T., Fei, S.: Exponential state estimation for recurrent nural networks with distributed delsys. Neurocomputing 71, 428–438 (2007)
6. Wang, Z., Liu, Y., Liu, X.: On global asymptotic stability of neural networks with discrete and distributed delays. Phys. Lett. A 345, 299–308 (2005)
7. Wang, Z., Shu, H., Liu, Y., Ho, D.W.C., Liu, X.: Robust stability analysis of generalized neural networks with discrete and distributed time delays. Chaos, Solitons Fractal 30, 886–896 (2006)
8. Liu, Y., Wang, Z., Liu, X.: Design of exponential state estimators for neural networks with mixed time delays. Phys. Lett. A 364, 401–412 (2007)
9. Park, J., Cho, H.: A delay-dependent asymptotic stability criterion of cellular neural networks with time-varying discrete and distributed delays. Chaos, Solitons and Fractals 33, 436–442 (2007)
10. He, Y., Wu, M.: Delay-dependent exponential stability of delayded neural networks with time-varying delay. IEEE Trans. Circuits Syst. 53, 553–556 (2006)
11. Xu, S., Lam, J.: Novel global stability criteria for interval neural networks with multiple time-varying delays. Phys. Lett. A 342, 322–330 (2005)
12. Liu, D., Zhang, J., Guan, X.: Exponential stability for neural networks: an LMI approach. Joural of Systems Engineering and Electronics 18, 68–71 (2007)

Mean Square Stability in the Numerical Simulation of Stochastic Delayed Hopfield Neural Networks with Markovian Switching

Hua Yang[1], Feng Jiang[2], and Jiangrong Liu[1]

[1] Department of Mathematics and Physics, Wuhan Polytechnic University, Wuhan 430023, China
[2] Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
{huay20,jeff20}@163.com, liujrjj@sohu.com

Abstract. This paper is concerned with the mean square stability for stochastic delayed Hopfield neural networks with Markovian switching. The sufficient conditions to guarantee the exponential stability in mean square of an equilibrium solution are given. Moreover, we give the mean square stability of the numerical method. The result shows that the numerical method shares the stability of the true solution.

Keywords: Hopfield neural networks, Markovian switching, Mean square stability.

1 Introduction

Stability of Hopfield neural networks plays an important role in their potential applications such as associative content-addressable memories, pattern recognition and optimization [1-5]. Recently, both delay independent and delay dependent sufficient conditions have been proposed to verify the asymptotical or exponential stability of delay systems.

In applications of neural networks, it is not uncommon for the parameters of neural networks change abruptly due to unexpected failure or designed switching [6]. In such a case, neural networks can be represented by a switching model which can be regarded as a set of parametric configurations switching from one to another according to a given Markov chain. Furthermore, in real nervous systems, the synaptic transmission is a noisy process brought on by random fluctuations from the release of neurotransmitters and other probabilistic causes. Hence, the stability analysis problem for stochastic neural networks with Markoving switching becomes increasingly significant, and some results related to this problem have recently been studied [5]. But few authors have considered the exponential stability for stochastic delayed systems with Markoving switching.

Most of stochastic Hopfield neural networks with Markovian switching, similar to stochastic differential equations with Markovian switching, do not have explicit solutions. Hence the numerical methods such as Euler-Maruyama scheme

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 577–584, 2010.

are used to deal with their properties. To best of our knowledge, there has been little work on the exponential stability of numerical methods for stochastic delay Hopfield neural networks, although there are many papers concerned with the numerical solutions to stochastic delay differential equations with Markovian switching [7-9].

This article is organized as follows. In Section 2, we give some notations. The sufficient conditions to guarantee the exponential stability in mean square of an equilibrium solution are given in Section 3. In Section 4 we will prove the exponential stability of the numerical method and give an example to illustrate our theory.

2 Preliminaries

Throughout this paper, unless otherwise specified, R^n and $R^{n\times m}$ denote, respectively, the n-dimensional Euclidean space and the set of $n \times m$ real matrices. Let $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, P)$ be a complete probability space with a filtration $\{\mathcal{F}_t\}_{t\geq 0}$ satisfying the usual conditions (i.e. the filtration contains all p-null sets and is right continuous). $B(t)$ be a scalar Brownian motion defined on the probability space. If A is a vector or matrix, its transpose is denoted by A^T. E denotes mathematics expectation with respect to P.

Let $r(t), t \geq 0$, be a right-continuous Markov chain on the probability space taking values in a finite state-space $S = \{1, 2, \cdots, N\}$ with generator $\Gamma = (\gamma_{ij})_{N\times N}$ given by

$$P\{r(t+\Delta) = j | r(t) = i\} = \begin{cases} \gamma_{ij}\Delta + o(\Delta), & \text{if} \quad i \neq j, \\ 1 + \gamma_{ii}\Delta + o(\Delta), & \text{if} \quad i = j, \end{cases} \tag{1}$$

where $\Delta > 0$. Here $\gamma_{ij} \geq 0$ is the transition rate from i to j if $i \neq j$. While $\gamma_{ii} = -\sum_{j\neq i} \gamma_{ij}$. We assume that the Markov chain $r(\cdot)$ is independent of the Brownian motion $B(t)$.

To analyze the Euler–Maruyama scheme as well as to simulate the approximate solution, we will need the following well-known lemma.

Lemma 2.1. *Given $\triangle > 0$, let $r_M^{\triangle} = r(M\triangle)$ for $M \geq 0$. Then $\{r_M^{\triangle}, M = 0, 1, 2, ...\}$ is a discrete Markov chain with the one-step transition probability matrix $P(\triangle) = (P_{ij}(\triangle))_{N\times N} = e^{\triangle\Gamma}$.*

Consider the stochastic delayed Hopfield neural networks with Markovian Switching

$$\begin{aligned} \mathrm{d}x(t) \;=\; & [-C(r(t))x(t) + A(r(t))f(x(t)) + B(r(t))g(x_\tau(t))]\mathrm{d}t \\ & +\sigma(x(t), r(t))\mathrm{d}B(t) \end{aligned} \tag{2}$$

with $x(t) = \xi(t), -\tau \leq t \leq 0$, where, we shall write $A(i) = A^i$ etc.

$x(t) = (x_1(t), x_2(t), \cdots, x_n(t))^T,\ C^i = \mathrm{diag}(C_1^i, C_2^i, \cdots, C_n^i),\ \tau = \max_{1\leq k\leq n}\tau_k,$

$A^i = (a_{kl}^i)_{n\times n},\ B^i = (b_{kl}^i)_{n\times n},\ f(x(t)) = (f_1(x_1(t)), \cdots, f_n(x_n(t)))^T,$

$g(x_\tau(t)) = (g_1(x_1(t-\tau_1)), \cdots, g_n(x_n(t-\tau_n)))^T$, $B(t) = (B_1(t), B_2(t), \cdots, B_n(t))^T$, $\xi(t) = (\xi_1(t), \cdots, \xi_n(t))^T, \sigma(x,i) = \sigma_k^i(x), x_\tau(t) = (x_1(t-\tau_1), \cdots, x_n(t-\tau_n))^T$.

In the above models, $n \geq 1$ is the number of neurons in the networks, x_k is the state variable of the ith neuron at time t, f_l and g_l denote the output of the lth unit at time t and $t-\tau_l$, respectively. $c_k^i, a_{kl}^i, b_{kl}^i$ and τ_l are constants: c_k^i represents the rate with which the kth unit will reset its potential to the resting state in isolation when disconnected from the network and the external stochastic perturbation in the mode of i, and is a positive constant; τ_l is the transmission delay along the axon of lth unit and is a nonnegative constant; a_{kl}^i and b_{kl}^i are weight the strength of the lth unit on the kth unit at time t and $t-\tau_l$ in the mode of i, respectively.

To study the stability of the solutions for Eq. (2), we need the following assumption:

(A1) $f_l(0) = g_l(0) = \sigma_k^i(0) = 0$. f_l, g_l and σ_k^i satisfy Lipschitz condition with Lipschitz constant $\alpha_l > 0, \beta_l > 0$ and $L_k^i > 0$, respectively.

It follows from [9] that under the assumption (A1), Eq. (2) has a global solution on $t \geq 0$ and any i_0, which is denoted by $x(t;\xi,i_0)$, or $x(t)$, if no confusion occurs. Clearly, Eq. (2) admits an equilibrium solution $x = 0$.

3 Exponential Stability

In the section we give the exponential stability of the true solution to Eq. (2).

Definition 3.1. *For Eq. (2) and every $\xi \in L^2_{\mathcal{F}_0}([-\tau,0];R^n), i_0 \in S$, the equilibrium point is exponentially stable in the mean square if for all network mode, there exist positive constants M and λ such that*

$$E|x(t)|^2 \leq Me^{-\lambda t}E|\xi|^2.$$

Theorem 3.1. *Eq. (2) is exponentially stable in mean square, if Eq. (2) satisfies (A1) and (A2): For each $i \in S, k = 1, 2, \cdots, n$,*

$$-2c_k^i + \sum_{l=1}^n |a_{kl}^i|\alpha_l + \sum_{l=1}^n |b_{kl}^i|\beta_l + \sum_{j=1}^N |\gamma_{ij}| + (L_k^i)^2 + \sum_{l=1}^n |a_{lk}^i|\alpha_k + \sum_{l=1}^n |b_{lk}^i|\beta_k < 0.$$

Proof. From the condition, there exists a sufficiently small positive constant λ such that

$$\begin{aligned} -2c_k^i + \sum_{l=1}^n |a_{kl}^i|\alpha_l \ + \ \sum_{l=1}^n |b_{kl}^i|\beta_l + \sum_{j=1}^N |\gamma_{ij}| + (L_k^i)^2 \\ + \sum_{l=1}^n |a_{lk}^i|\alpha_k + e^{\lambda\tau}\sum_{l=1}^n |b_{lk}^i|\beta_k \leq 0 \end{aligned}$$

We define a Lyapunvon function as $V(x,t,i) = e^{\lambda t}x^T(t)x(t)$, applying Itô formula along Eq. (2), we can have

$$\begin{aligned}
V(x(t),t,i) &= V(x(0),0,i_0) + \int_0^t \lambda e^{\lambda s}x^T(s)x(s)\mathrm{d}s + \int_0^t e^{\lambda s}\sum_{j=1}^{N}\gamma_{ij}x^T(s)x(s)\mathrm{d}s \\
&\quad + \int_0^t 2e^{\lambda s}\{-C^i x(s) + A^i f(x(s)) + B^i g(x_\tau(s))\}\mathrm{d}s \\
&\quad + \int_0^t e^{\lambda s}\mathrm{trace}[\sigma^T(x(s),i)\sigma(x(s),i)]\mathrm{d}s + M(t) \\
&\le V(x(0),0,i_0) + \int_0^t \lambda e^{\lambda s}x^T(s)x(s)\mathrm{d}s + M(t) \\
&\quad + \int_0^t 2e^{\lambda s}\sum_{k=1}^{n}\Big\{\big[-c_k^i + \frac{1}{2}\sum_{l=1}^{n}|a_{kl}^i|\alpha_l + \frac{1}{2}\sum_{l=1}^{n}|b_{kl}^i|\beta_l \\
&\quad + \frac{1}{2}\sum_{l=1}^{n}|\gamma_{ij}|\big]x_k^2(s) + [\frac{1}{2}\sum_{l=1}^{n}|a_{kl}^i|\alpha_l + \frac{1}{2}\sum_{l=1}^{n}(L_l^i)^2]x_l^2(s) \\
&\quad + \frac{1}{2}\sum_{l=1}^{n}|b_{kl}^i|\beta_l x_l^2(s-\tau_l)\Big\}\mathrm{d}s,
\end{aligned}$$

where $M(t) = \int_0^t 2e^{\lambda s}x^T(s)\sigma(x(s))\mathrm{d}B(s)$. On the other hand, for $t > 0$, it is easy to see that

$$\int_0^t e^{\lambda s}x_l^2(s-\tau_l)\mathrm{d}s \le e^{\lambda\tau}\int_{-\tau}^0 e^{\lambda s}x_l^2(s)\mathrm{d}s + e^{\lambda\tau}\int_0^t e^{\lambda s}x_l^2(s)\mathrm{d}s.$$

Consequently,

$$\begin{aligned}
V(x(t),t,i) &\le V(x(0),0,i_0) + M(t) + e^{\lambda\tau}\int_{-\tau}^0 e^{\lambda s}\sum_{k=1}^{n}\sum_{l=1}^{n}|b_{kl}^i|\beta_l|x_l(s)|^2\mathrm{d}s \\
&\quad + \int_0^t e^{\lambda s}\sum_{k=1}^{n}\Big\{\big[\lambda - 2c_k^i + \sum_{l=1}^{n}|a_{kl}^i|\alpha_l + \sum_{l=1}^{n}|b_{kl}^i|\beta_l + \sum_{l=1}^{N}|\gamma_{ij}| \\
&\quad + (L_k^i)^2 + \sum_{l=1}^{n}|a_{lk}^i|\alpha_k + e^{\lambda\tau}\sum_{l=1}^{n}|b_{lk}^i|\beta_k\big]x_k^2(s)\Big\}\mathrm{d}s \\
&\le V(x(0),0,i_0) + M(t) + e^{\lambda\tau}\int_{-\tau}^0 e^{\lambda s}\sum_{k=1}^{n}\sum_{l=1}^{n}|b_{kl}^i|\beta_l|x_l(s)|^2\mathrm{d}s
\end{aligned}$$

We hence obtain that

$$Ee^{\lambda t}|x(t)|^2 \le E|x(0)|^2 + e^{\lambda\tau}\int_{-\tau}^0 Ee^{\lambda s}\sum_{k=1}^{n}\sum_{l=1}^{n}|b_{kl}^i|\beta_l|x_l(s)|^2\mathrm{d}s.$$

This completes the proof.

4 Stability in the Numerical Simulation

In this section we shall analyze the stability of the numerical method for Eq. (2). Now apply the Euler–Maruyama method to Eq. (2), then we have

$$\begin{aligned} y_{k,M+1} &= y_{k,M} + [-c_k(r_M^{\triangle})y_{k,M} + \sum_{l=1}^{n} a_{kl}(r_M^{\triangle})f_l(y_{k,M}) \\ &\quad + \sum_{l=1}^{n} b_{kl}(r_M^{\triangle})g_l(y_{k,M-m_l})]\triangle + \sigma_k(y_{k,M}, r_M^{\triangle})\triangle B_{k,M}. \end{aligned} \tag{3}$$

where $\triangle$ is a stepsize which satisfied $\tau_l = m_l\triangle$ for a positive integer m_l and $t_M = M\triangle$. When $t_M \le 0$, we have $y_{k,M} = \xi_k(t_M)$. Moreover, $\triangle B_{k,M}$ is normal distribution with mean zero and variance $\triangle$.

Definition 4.1. Under condition (A1), (A2) and (A3): For each $i \in S$,

$$\sum_{l=1}^{n} |a_{kl}^i|\alpha_l + \sum_{l=1}^{n} |b_{kl}^i|\beta_l \le \sum_{l=1}^{n} |a_{lk}^i|\alpha_k + \sum_{l=1}^{n} |b_{lk}^i|\beta_k.$$

Then a numerical method is said to be mean square stable, if there exist a $\triangle > 0$ such that the numerical solution sequence $\{y_{k,M}\}$ produced by this numerical scheme satisfies $\lim_{M\to\infty} E|Y_{k,M}|^2 = 0$. for every stepsize $\triangle \in (0, \triangle_0)$ with $\triangle = \frac{\tau_l}{m_l}$.

Theorem 4.1. *Assume that for any $i \in S$, (A1)−(A3), then the Euler–Maruyama method for Eq. (2) is mean square stable.*

Proof. By Lemma 2.1, the generation of $r_M^{\triangle}$ occurs before computing $y_{k,M+1}$, then $r_M^{\triangle}$ is known. Since $r_M^{\triangle} \in S$, for any $i \in S$, from (3), we then have

$$\begin{aligned} y_{k,M+1} &= [(1 - c_k^i\triangle)y_{k,M} + \sigma_k(y_{k,M}, i)\triangle B_{k,M}] \\ &\quad + \sum_{l=1}^{n} a_{kl}^i f_l(y_{k,M})\triangle + \sum_{l=1}^{n} b_{kl}^i g_l(y_{k,M-m_l})\triangle. \end{aligned}$$

Squaring both sides of the above equality, we obtain

$$\begin{aligned} y_{k,M+1}^2 &= [(1 - c_k^i\triangle)y_{k,M} + \sigma_k(y_{k,M}, i)\triangle B_{k,M}]^2 + \triangle^2(\sum_{l=1}^{n} a_{kl}^i f_l(y_{k,M}))^2 \\ &\quad + 2\triangle[(1 - c_k^i\triangle)y_{k,M} + \sigma_k(y_{k,M}, i)\triangle B_{k,M}] \sum_{l=1}^{n} a_{kl}^i f_l(y_{k,M}) \\ &\quad + 2\triangle[(1 - c_k^i\triangle)y_{k,M} + \sigma_k(y_{k,M}, i)\triangle B_{k,M}] \sum_{l=1}^{n} b_{kl}^i g_l(y_{k,M-m_l}) \\ &\quad + 2\triangle^2 \sum_{l=1}^{n} a_{kl}^i f_l(y_{k,M}) \sum_{l=1}^{n} b_{kl}^i g_l(y_{k,M-m_l}) + \triangle^2(\sum_{l=1}^{n} b_{kl}^i g_l(y_{k,M-m_l}))^2. \end{aligned}$$

It follow from $2abxy \le |ab|(x^2+y^2), a, b \in R$, that

$$\begin{aligned} y_{k,M+1}^2 \le\ & 2(1-c_k^i\triangle)y_{k,M}\sigma_k(y_{k,M},i)\triangle B_{k,M} \\ & +(1-c_k^i\triangle)^2 y_{k,M}^2 + \sigma_k^2(y_{k,M},i)(\triangle B_{k,M})^2 \\ & +\triangle^2\sum_{l=1}^n |a_{kl}^i|\alpha_l \sum_{p=1}^n |a_{kp}^i|\alpha_p y_{k,M}^2 + \triangle^2 \sum_{l=1}^n |b_{kl}^i|\beta_l \sum_{p=1}^n |b_{kp}^i|\beta_p y_{k,M-m_l}^2 \\ & +\triangle\sum_{l=1}^n |(1-c_k^i\triangle)a_{kl}^i|\alpha_l [y_{k,M}^2 + y_{l,M}^2] \\ & +2\triangle\sigma_k^2(y_{k,M},i)\triangle B_{k,M}\sum_{l=1}^n a_{kl}^i f_l(y_{k,M}) \\ & +\triangle\sum_{l=1}^n |(1-c_k^i\triangle)b_{kl}^i|\beta_l [y_{k,M}^2 + y_{l,M-m_l}^2] \\ & +2\triangle\sigma_k^2(y_{k,M},i)\triangle B_{k,M}\sum_{l=1}^n b_{kl}^i g_l(y_{k,M-m_l}) \\ & +\triangle^2\sum_{l=1}^n |a_{kl}^i|\alpha_l \sum_{l=1}^n |a_{kl}^i|\beta_l [y_{k,M}^2 + y_{l,M-m_l}^2]. \end{aligned} \tag{4}$$

Note that $E\triangle B_{k,M} = 0, E(\triangle B_{k,M})^2 = \triangle$ and $f_l(y_{k,M}), g_l(y_{k,M-m_l}), \sigma_k(y_{k,M},i)$ are $\mathcal{F}_{t_M}$-measurable. Consequently,

$$E[\triangle B_{k,M} f_l(y_{k,M})\sigma_k(y_{k,M},i)] = E[\triangle B_{k,M} g_l(y_{k,M-m_l})\sigma_k(y_{k,M},i)] = 0$$

and

$$E[(\triangle B_{k,M})^2\sigma_k(y_{k,M},i)] = \triangle E\sigma_k(y_{k,M},i)^2.$$

Let $Y_{k,M} = Ey_{k,M}^2$, From (4) and (A1), we hence have

$$Y_{k,M+1} \le PY_{k,M} + \sum_{l=1}^n Q_l Y_{l,M} + \sum_{l=1}^n R_l Y_{l,M-m_l}, \tag{5}$$

where

$$\begin{aligned} P &= (1-c_k^i\triangle)^2 + (L_k^i)^2\triangle + \triangle\sum_{l=1}^n |(1-c_k^i\triangle)a_{kl}^i|\alpha_l + \triangle\sum_{l=1}^n |(1-c_k^i\triangle)b_{kl}^i|\beta_l, \\ Q_l &= \triangle^2|a_{kl}^i|\alpha_l\sum_{p=1}^n |a_{kp}^i|\alpha_p + \triangle|(1-c_k^i\triangle)a_{kl}^i|\alpha_l + \triangle^2|a_{kl}^i|\alpha_l\sum_{l=1}^n |b_{kp}^i|\beta_p, \\ R_l &= \triangle^2|b_{kl}^i|\beta_l\sum_{p=1}^n |b_{kp}^i|\beta_p + \triangle|(1-c_k^i\triangle)b_{kl}^i|\beta_l + \triangle^2|a_{kl}^i|\alpha_l\sum_{l=1}^n |b_{kp}^i|\beta_p. \end{aligned}$$

Then $Y_{k,M+1} \le (P + \sum_{l=1}^n Q_l + \sum_{l=1}^n R_l)\max_{1\le l\le n}\{Y_{k,M}, Y_{l,M}, Y_{l,M-m_l}\}$. By recursive calculation we conclude that $Y_{k,M} \to 0 (M \to \infty)$ is $P + \sum_{l=1}^n Q_l +$

$\sum_{l=1}^{n} R_l < 1$, which is equivalent to

$$-2c_k^i + (L_k^i)^2 + 2\sum_{l=1}^{n}[|1 - c_k^i \triangle a_{kl}^i|\alpha_l + |1 - c_k^i \triangle b_{kl}^i|\beta_l]$$
$$+\triangle\Big[(c_k^i)^2 + (\sum_{l=1}^{n}|a_{kl}^i|\alpha_l)^2 + (\sum_{l=1}^{n}|b_{kl}^i|\beta_l)^2 + 2\sum_{l=1}^{n}|a_{kl}^i|\alpha_l \sum_{l=1}^{n}|b_{kl}^i|\beta_l\Big] < 0. \quad (6)$$

Let

$$\triangle_0' = \min\left\{\min_{i\in S}\frac{1}{|c_k^i|}, \min_{i\in S}\frac{2c_k^i - (L_k^i)^2 - 2\sum_{l=1}^{n}|a_{kl}^i|\alpha_l - 2\sum_{l=1}^{n}|b_{kl}^i|\beta_l}{(c_k^i - \sum_{l=1}^{n}|a_{kl}^i|\alpha_l - \sum_{l=1}^{n}|b_{kl}^i|\beta_l)^2}\right\}.$$

By (A2) and (A3), we have $\triangle_0' > 0$. If $\triangle \in (0, \triangle_0')$, we have

$$-2c_k^i + (L_k^i)^2 + 2\sum_{l=1}^{n}|a_{kl}^i|\alpha_l + 2\sum_{l=1}^{n}|b_{kl}^i|\beta_l + (c_k^i - \sum_{l=1}^{n}|a_{kl}^i|\alpha_l - \sum_{l=1}^{n}|b_{kl}^i|\beta_l)^2 < 0,$$

which shows that (6) holds. Now let $\triangle_0 = \min\{1, \triangle_0'\}$, then $\lim_{M\to\infty} E|y_{k,M}|^2 = 0$. This proof is complete.

To illustrate our result, we give an example as follows. Let $B(t)$ be a Brown motion. Let $r(t)$ be a right-continuous Markov chain taking values in S={1,2} with generator

$$\Gamma = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Assume that $B(t)$ and $r(t)$ are independent. Consider the following equation

$$\mathrm{d}x(t) = [-C^i x(t) + A^i f(x(t)) + B^i g(x_\tau(t))]\mathrm{d}t + \sigma^i(x(t))\mathrm{d}B(t), \quad (7)$$

with the initial segment $x_1(t) = t+1, t \in [-1, 0], r(0) = 1$, where $C^1 = -9, C^2 = -0.1, A^1 = 1, A^2 = 1, B^1 = 1, B^2 = 4, \sigma^1(x) = -2x, \sigma^2(x) = -9x, f(x) = \sin x, \quad g(x) = x$. It is obvious that $\alpha_1 = \beta_1 = 1$, and the conditions of Theorem

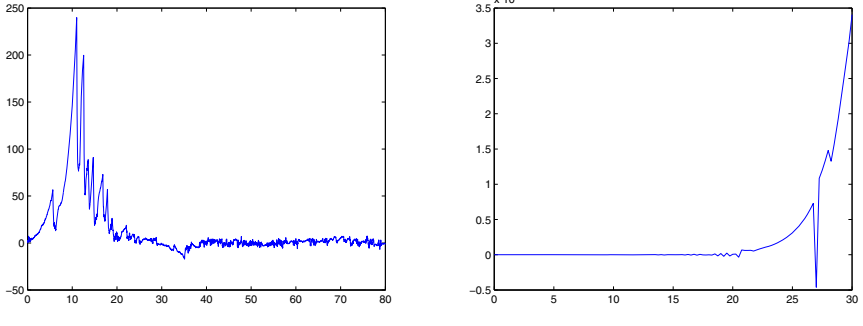


Fig. 1. Numerical simulation: left $\triangle = 0.02$ and right $\triangle = 0.25$

4.1 hold. Hence the numerical method is mean square stable if $\triangle \in (0, 0.11)$. The data used in all figures are obtained by the mean-square of data by 100 trajectories, that is, $\omega_i : 1 \leq i \leq 100, Y_n = \frac{1}{100} \sum_{i=1}^{100} |Y_n(\omega_i)|^2$. For this example, Fig. 1 reveals the numerical solution will tend to 0 on $\triangle = 0.02$ but the numerical solution will not tend to 0 on $\triangle = 0.25$.

5 Conclusions

In this paper, we investigate the mean square exponential stability for stochastic delayed Hopfield neural networks with Markovian switching. Notice that the criteria on stochastic delayed Hopeld neural networks with Markovian switching are independent of the delays, which means that the delays are harmless in stochastic delayed Hopeld neural networks with Markovian switch- ing with the structure satisfying the conditions. Moreover, we give the square mean stability of the numerical method to Eq. (2). The result shows that the numerical method shares the stability of the true solution.

References

1. Chen, T.: Global Exponential Stability of Delayed Hopfield Neural Networks. Neural Netw. 14, 977–980 (2001)
2. Guo, S., Huang, L.: Stability Analysis of a Delayed Hopfield Neural Network. Phys. Rev. E 67, 061902 (2003)
3. Huang, C., Chen, P., He, Y., Huang, L., Tan, W.: Almost Sure Exponential Stability of Delayed Hopfield Neural Networks. Applied Mathematics Letters 21, 701–705 (2008)
4. Huang, H., Ho, D.W.C., Qu, Y.: Robust Stability of Stochastic Delayed Additive Neural Networks with Markovian Switching. Neural Netw. 20, 799–809 (2007)
5. Wan, L., Sun, J.: Mean Square Exponential Stability of Stochastic Delayed Hopfield Neural Networks. Phys. Lett. A 343, 306–318 (2005)
6. Liberzon, D.: Switching in Systems and Control. Birkhauser, Boston (2003)
7. Liu, M.Z., Cao, W.R., Fan, Z.C.: Convergence and Stability of the Semi-implicit Euler Method for a Linear Stochastic Differential Delay Equation. Journal of Computational and Applied Mathematics 170, 255–268 (2004)
8. Rathinasamy, A., Balachandran, K.: Mean Square Stability of Semi-implicit Euler Method for Linear Stochastic Differential Equations with Multiple Delays and Markovian Switching. Applied Mathematics and Computation 206, 968–979 (2008)
9. Mao, X., Yuan, C.: Stochastic Differential Equations with Markovian Switching. Imperial College Press, London (2006)

The Existence of Anti-periodic Solutions for High-Order Cohen-Grossberg Neural Networks

Zhouhong Li, Kaihong Zhao*, and Chenxi Yang

Department of Mathematics, Yuxi Normal University
Yuxi, Yunnan 653100, China
zhaokaihongs@126.com

Abstract. In this paper, we use the Lyapunov function to establish new results on the existence uniqueness and globally exponential stablity of anti-periodic solutions for high-order Cohen-Grossberg neural networks with time-varying delays. Finally, an example and its simulation are given to illustrate the feasibility and effectiveness of our results.

Key words: High-order Cohen-Grossberg neural networks, Anti-periodic solution, Exponential stability, Delay.

1 Introduction

Since Cohen-Grossberg neural networks (CGNNs) was first proposed by Cohen and Grossberg [1] in 1983, many authors have done extensive works on this subject due to their applications in many fields such as pattern recognition, parallel computing, associative memory, signal and image processing and combinatorial optimization. In such applications, it is of prime importance to ensure that the designed neural networks is stable.

In reality, high-order neural networks have stronger approximation property, faster convergence rate, greater storage capacity, and higher fault tolerance than lower-order neural networks. High-order neural networks have been the object of intensive analysis by numerous authors in recent years. In particular, there have been extensive results on the problem of the existence and stability of equilibrium points and periodic solutions of CGNNs in the literature. We refer the reader to [1-11] and the references cited therein. To the best of our knowledge, few authors have considered anti-periodic solutions for high-order Cohen-Grossberg neural networks. This motivates us to consider the anti-periodic solution for the following high-order Cohen-Grossberg neural networks with time-varying delays described by

$$x_i'(t) = -a_i(x_i(t))[b_i(x_i(t)) - \sum_{j=1}^{n} c_{ij}(t)f_j(x_j(t)) - \sum_{j=1}^{n} d_{ij}(t)g_j(x_j(t-\tau_{ij}(t)))$$
$$- \sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t)h_j(x_j(t-\sigma_{ijl}(t)))h_l(x_l(t-\nu_{ijl}(t))) - I_i(t)], \quad (1)$$

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 585–594, 2010.

where $i, j = 1, 2, \ldots, n$, and n corresponds to the number of units in the neural network. $x_i(t)$ corresponds to the state of the ith unit at time t. a_i represents an amplification function. b_i is an appropriately behaved function. $c_{ij}(t)$ denotes the strengths of connectivity between cell i and j at time t. $d_{ij}(t)$ and $e_{ijl}(t)$ are the first-order and second-order connection weights of the Cohen-Grossberg neural network. $\tau_{ij}(t) > 0, \sigma_{ijl}(t) > 0$ and $\nu_{ijl}(t) > 0$ correspond to the transmission delays, respectively. $I_i(t)$ denotes the external inputs at time t. f_j, g_j and h_j are the activation function of signal transmission.

A primary purpose of this paper is to give the conditions for the existence and exponential stability of anti-periodic solutions for system (1). We will establish some sufficient conditions for the existence and uniqueness and globally exponential stability of anti-periodic solutions of system (1). Our results different from those of the references listed above.

Let $u(t) : R \to R$ be continuous in t. $u(t)$ is said to be T-anti-periodic on R , if $u(t+T) = -u(t)$, for all $t \in R$. If a system is T-anti-periodic $x(t+T) = -x(t)$, then it is $2T$-periodic $x(t+2T) = -x(t+T) = x(t)$.

Denote $R_+ = [0, +\infty)$. Throughout this paper, for $i, j, l = 1, 2, \ldots, n$ and for all $t, u \in R$, it will be assumed that $a_i,\ \tau_{ij},\ \sigma_{ijl},\ v_{ijl} \in C(R, R_+)$ and $b_i, c_{ij}, d_{ij}, e_{ijl}, I_i \in C(R, R)$ and c_{ij}, d_{ij} and e_{ijl} are $2T$-periodic, and

$$a_i(-u) = a_i(u),\ \ b_i(-u) = -b_i(u),\ \ c_{ij}(t+T) = c_{ij}(t), \tag{2}$$

$$d_{ij}(t+T)g_j(u) = -d_{ij}(t)g_j(-u),\quad I_i(t+T) = -I_i(t), \tag{3}$$

$$e_{ijl}(t+T)h_j(u)h_l(u) = -e_{ijl}(t)h_j(-u)h_l(-u), \tag{4}$$

$$\tau_{ij}(t+T) = \tau_{ij}(t),\ \sigma_{ijl}(t+T) = \sigma_{ijl}(t),\ \nu_{ijl}(t+T) = \nu_{ijl}(t). \tag{5}$$

Morever, we suppose that there exits constant τ such that

$$\tau = \max\{\max_{1\le i,j\le n} \sup_{t\in R} \tau_{ij}(t), \max_{1\le i,l\le n} \sup_{t\in R} \sigma_{ijl}(t), \max_{1\le i,l\le n} \sup_{t\in R} v_{ijl}(t)\}.$$

We also assume that the following conditions hold.

(H1) there exists a positive constant $\overline{a}_i$ such that $a_i(u) \le \overline{a}_i$, for all $u \in R, i = 1, 2, \ldots, n$;

(H2) $b_i(0) = 0$ and there exist positive constants L_i^a and L_i^{ab} such that $|a_i(u) - a_i(v)| \le L_i^a|u - v|$, $|a_i(u)b_i(u) - a_i(v)b_i(v)| \le L_i^{ab}|u - v|$, for all $u, v \in R, i = 1, 2, \ldots, n$;

(H3) for each $j \in \{1, 2, \ldots, n\}$, there exist nonnegative constants L_j^f, L_j^g and L_j^h such that

$$f_j(0) = 0, |f_j(u) - f_j(v)| \le L_j^f|u - v|, g_j(0) = 0, |g_j(u) - g_j(v) \le L_j^g|u - v|,$$
$$h_j(0) = 0, |h_j(u) - h_j(v)| \le L_j^h|u - v|, |h_j(u)| \le M_j^h, \quad \text{for all } u, v \in R.$$

(H4) there exists an constant $\beta > 0$ such that for all $t > 0$, $H < -\beta < 0$.

For convenience, we introduce some notations. We will use $x(t) = (x_1(t), \ldots, x_n(t))^T \in R^n$ to denote a column vector in which they symbol $()^T$ denotes the transpose of a vector. We let $|x|$ denote absolute-value vector given by $|x| = (|x_1|, |x_2|, \ldots, |x_n|)^T$, and define $||x|| = \max_{1\le i\le n} |x_i|$.

The initial conditions associated with system (1) are of the form $x_i(s) = \varphi_i(s), \quad s \in (-\tau, 0], \ i = 1, 2, \ldots, n$, where $\varphi = (\varphi_1(t), \varphi_2(t), \ldots, \varphi_n(t))^T \in C([-\tau, 0]; R^n)$.

Definition 1. *Let $x^*(t) = (x_1^*(t), x_2^*(t), \ldots, x_n^*(t))^T$ be an anti-periodic solution of system (1) with initial value $\varphi^*(t) = (\varphi_1^*(t), \ldots,$
$\varphi_n^*(t))^T$. If there exist constants $\lambda > 0$ and $M > 1$ such that for every solution $x(t) = (x_1(t), x_2(t), \ldots, x_n(t))^T$ of system (1) with any initial value $\varphi(t) = (\varphi_1(t), \varphi_2(t), \ldots, \varphi_n(t))^T$,*

$$|x_i(t) - x_i^*(t)| \leq M||\varphi - \varphi^*||_\infty e^{-\lambda t}, \ \forall t > 0, i = 1, 2, \ldots, n,$$

where $\|\varphi - \varphi^*\|_\infty = \sup_{-\infty \leq s \leq n} \max_{1 \leq i \leq n} |\varphi_i(s) - \varphi_i^*(s)|$, then $x^*(t)$ is said to be globally exponentially stable.

The organization of the rest of this paper is as follows. In Section 2, we introduce some definitions and make some preparations for later sections. In Section 3, we establish our main results for the existence and exponential stability of anti-periodic solutions of (1). Finally, we present an example to illustrate the feasibility and effectiveness of our results obtained in previous sections.

2 Preliminary Results

The following lemmas will be used to prove our main results in Section 3.

Lemma 1. *Let $(H1)-(H4)$ hold. Suppose that $\tilde{x}(t) = (\tilde{x}_1(t), \tilde{x}_2(t), \ldots, \tilde{x}_n(t))^T$ is a solution of system (1) with initial conditions*

$$\tilde{x}_i(s) = \tilde{\varphi}_i(s), \quad |\tilde{\varphi}_i(s)| < \gamma, s \in [-\tau, 0], i = 1, 2, \ldots, n, \tag{6}$$

where

$$\gamma > (L_i^{ab})^{-1}\overline{a_j}\Bigg[\sum_{j=1}^{n}|c_{ij}(t)|L_j^f + \sum_{j=1}^{n}|d_{ij}(t)|M_j^g + \sum_{j=1}^{n}\sum_{l=1}^{n}|e_{ijl}(t)|M_j^h M_l^h + |I_i(t)|\Bigg] \tag{7}$$

then

$$|x_i(t)| < \gamma, \text{ for all } t > 0, \ \ i = 1, 2, \ldots, n. \tag{8}$$

Proof. Assume, by way of contradiction, that (8) does not hold. Then, there must exist $i \in \{1, 2, \ldots, n\}$ and $t_0 > 0$ such that

$$|\tilde{x}_i(t_0)| = \gamma \text{ and } |\tilde{x}_j(t)| < \gamma, \text{ for all } t \in (-\infty, t_0], j = 1, 2, \ldots, n. \tag{9}$$

Calculating the upper left derivative of $|\tilde{x}_i(t)|$, together with $(H1)-(H4)$, (8) implies that

$$\begin{aligned}
0 \le D^+(|\tilde{x}_i(t_0)|) &\le -L_i^{ab}\tilde{x}_i(t_0) + \overline{a_j}|\sum_{j=1}^{n} c_{ij}(t_0)L_j^f\tilde{x}_i(t_0)\sum_{j=1}^{n} d_{ij(t_0)}g_j(\tilde{x}_j(t_0 \\
&\quad -\tau_{ij}(t_0))) + \sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t_0)h_j(t_0-\sigma_{ijl}(t_0))h_l(x_l(t_0-\nu_{ijl(t_0)}))) + I_i(t_0)| \\
&\le -L_i^{ab}\gamma + (\sum_{j=1}^{n}|c_{ij}(t_0)|L_j^f + \sum_{j=1}^{n}|d_{ij}(t_0)|M_j^g + \sum_{j=1}^{n}\sum_{l=1}^{n}|e_{ijl}(t_0)|M_j^hM_l^h \\
&\quad +|I_i(t_0)|)\overline{a_j} < 0,
\end{aligned}$$

Which is a contradiction and implies that (8) holds, the proof of Lemma 1 is now completed.

Remark 1. In view of the boundedness of this solution, from the theory of functional differential equations in [13], it follows that $\widetilde{x}(t)$ on $[-\tau,\infty)$, provided that initial conditions are bounded by γ.

Lemma 2. *Suppose that* $(H1)-(H4)$ *are satisfied. Let* $x^*(t) = (x_1^*(t), x_2^*(t),\ldots,x_n^*(t))^T$ *be the solution of system (1) with initial value* $\varphi^*(t) = (\varphi_1^*(t), \varphi_2^*(t),\ldots,\varphi_n^*(t))^T, ||\varphi^*(t)|| < \gamma$, *and* $x(t) = (x_1(t), x_2(t),\ldots,x_n(t))^T$ *be the solution of system (1) with initial value* $\varphi(t) = (\varphi_1(t),\varphi_2(t),\ldots,\varphi_n(t))^T$. *Then there exist constants* $\lambda > 0$ *and* $M_\varphi > 1$ *such that*

$$|x_i(t) - x_i^*(t)| \le M_\varphi||\varphi-\varphi^*||_\infty e^{-\lambda t}, \quad \text{for all } t>0, \quad i=1,2,\ldots,n.$$

Proof. Let $y(t) = x(t) - x^*(t)$. Then

$$\begin{aligned}
y_i'(t) &= -[a_i(x_i(t))b_i(x_i(t)) - a_i(x_i^*(t))b_i(x_i^*(t))] + \sum_{j=1}^{n} c_{ij}(t)a_i(x_i(t))[f_j(x_j(t)) \\
&\quad -f_j(x_j^*(t))] + \sum_{j=1}^{n} c_{ij}(t)[a_i(x_i(t)) - a_i(x_i^*(t))]f_j(x_j^*(t)) + \sum_{j=1}^{n} a_i(x_i(t)) \\
&\quad \times d_{ij}(t)[g_j(x_j(t-\tau_{ij}(t))) - g_j(x_j^*(t-\tau_{ij}(t)))] + \sum_{j=1}^{n} d_{ij}(t)[a_i(x_i(t)) \\
&\quad -a_i(x_i^*(t))]g_j(x_j^*(t-\tau_{ij}(t))) + \sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t)a_i(x_i(t))[h_j(x_j(t-\sigma_{ijl}(t))) \\
&\quad \times h_l(x_l(t-\nu_{ijl}(t))) - h_j(x_j^*(t-\sigma_{ijl}(t)))h_l(x_l^*(t-\nu_{ijl}(t)))] \\
&\quad + \sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t)[a_i(x_i(t)) - a_i(x_i^*(t))]h_j(x_j^*(t-\sigma_{ijl}(t)))h_l(x_l^*(t-\nu_{ijl}(t))) \\
&\quad +[a_i(x_i(t)) - a_i(x_i^*(t))]I_i(t), i=1,2,\ldots,n. \qquad (10)
\end{aligned}$$

We consider the Lyapunov functional

$$V_i(t) = |y_i(t)|e^{\lambda t}, i = 1, 2, \ldots, n. \tag{11}$$

Calculating the left upper derivative of $V_i(t)$ along the solution $y(t)$ of system (10) with the initial value $\bar{\varphi} = \varphi - \varphi^*$, form (10) and (11), we have

$$\begin{aligned}
D^+(V_i(t)) \leq & -L_i^{ab}|y_i(t)|e^{\lambda t} + \lambda|y_i(t)|e^{\lambda t} + \sum_{j=1}^{n}|c_{ij}(t)|\overline{a_i}L_j^f|y_i(t)|e^{\lambda t} \\
& + \sum_{j=1}^{n}|c_{ij}(t)|L_j^a|y_i(t)|e^{\lambda t}L_j^f\gamma + \sum_{j=1}^{n}|d_{ij}(t)|\overline{a_i}L_j^g|y_j(t-\tau_{ij}(t))|e^{\lambda t} \\
& + \sum_{j=1}^{n}|d_{ij}(t)|\overline{a_i}|y_i(t)|e^{\lambda t}L_j^g\gamma + \sum_{j=1}^{n}\sum_{l=1}^{n}[\overline{a_i}|e_{ijl}(t)|(L_j^hM_l^h|y_j(t \\
& -\sigma_{ijl}(t))|e^{\lambda t} + M_j^hL_l^h|y_l(t-\nu_{ijl}(t))|)]e^{\lambda t} + \sum_{j=1}^{n}\sum_{l=1}^{n}[|e_{ijl}(t)|L_i^a|y_i(t)| \\
& \times e^{\lambda t}M_j^hM_l^h + L_i^aI_i(t)|y_i(t)|e^{\lambda t} \\
\leq & \Bigg[\lambda - L_i^{ab} + \sum_{j=1}^{n}|c_{ij}(t)|\overline{a_i}L_j^f + \sum_{j=1}^{n}|c_{ij}(t)|L_j^aL_j^f\gamma + \sum_{j=1}^{n}|d_{ij}(t)|\overline{a_i}L_j^g\gamma \\
& + \sum_{j=1}^{n}\sum_{l=1}^{n}|e_{ijl}(t)|L_i^aM_j^hM_l^h + L_i^aI_i(t)\Bigg]|y_i(t)|e^{\lambda t} + \Bigg[\sum_{j=1}^{n}|d_{ij}(t)|\overline{a_i}L_j^g \\
& \times|y_j(t-\tau_{ij}(t))| + \sum_{j=1}^{n}\sum_{l=1}^{n}\overline{a_i}|e_{ijl}(t)|(L_j^hM_l^h|y_j(t-\sigma_{ijl}(t))| + M_j^hL_l^h \\
& \times|y_l(t-\nu_{ijl}(t))|)\Bigg]e^{\lambda t}, i, j, l = 1, 2, \ldots, n.
\end{aligned} \tag{12}$$

Let $M > 1$ denote an arbitrary real number such that

$$M||\varphi - \varphi^*||_\infty = \sup_{-\infty<s\leq 0}\max_{1\leq j\leq n}|\varphi_j(s) - \varphi_j^*(s)| > 0, i = 1, 2, \ldots, n.$$

It follows from (11) that

$$V_i(t) = |y_i(t)|e^{\lambda t} < M||\varphi - \varphi^*||_\infty, \text{for all } t \in [-\tau, 0], i = 1, 2, \ldots, n. \tag{13}$$

We claim that

$$V_i(t) = |y_i(t)|e^{\lambda t} < M||\varphi - \varphi^*||_\infty, \text{for all } t > 0, i = 1, 2, \ldots, n. \tag{14}$$

Otherwise, there must exist $i, j \in \{1, 2, \ldots, n\}$ and $t_i > 0$ such that

$$V_i(t_i) = M||\varphi - \varphi^*||_\infty \; and \; V_j(t) < M||\varphi - \varphi^*||_\infty, \;\; \forall t \in [-\tau, t_i). \tag{15}$$

It follows from (11) that

$$V_i(t_i) - M||\varphi - \varphi^*||_\infty = 0 \; and \; V_j(t) - M||\varphi - \varphi^*||_\infty < 0, \; \forall t \in [-\tau, t_i). \quad (16)$$

Together with (12) and (16), we obtain

$$\begin{aligned}
0 &\le D^+(V_i(t_i) - M||\varphi - \varphi^*||_\infty) = D^+(V_i(t_i)) \\
&\le \Bigg[\lambda - L_i^{ab} + \sum_{j=1}^{n} |c_{ij}(t_i)|\overline{a_i}L_j^f + \sum_{j=1}^{n} |c_{ij}(t_i)|L_j^a L_j^f \gamma + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g \gamma \\
&\quad + \sum_{j=1}^{n}\sum_{l=1}^{n} |e_{ijl}(t_i)|L_j^a M_j^h M_l^h + L_i^a I_i(t_i)\Bigg] |y_i(t_i)|e^{\lambda t_i} + \Bigg[\sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g |y_j(t_i \\
&\quad - \tau_{ij}(t_i))| + \sum_{j=1}^{n}\sum_{l=1}^{n} \overline{a_i}|e_{ijl}(t_i)|(L_j^h M_l^h |y_j(t_i - \sigma_{ijl}(t_i))| + M_j^h L_l^h |y_l(t_i \\
&\quad - v_{ijl}(t_i))|)\Bigg] e^{\lambda t_i} \\
&= \Bigg[\lambda - L_i^{ab} + \sum_{j=1}^{n} |c_{ij}(t_i)|\overline{a_i}L_j^f + \sum_{j=1}^{n} |c_{ij}(t_i)|L_j^a L_j^f \gamma + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g \gamma \\
&\quad + \sum_{j=1}^{n}\sum_{l=1}^{n} |e_{ijl}(t_i)|L_i^a M_j^h M_l^h + L_i^a I_i(t_i)\Bigg] |y_i(t_i)|e^{\lambda t_i} + \Bigg[\sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g |y_j(t_i \\
&\quad - \tau_{ij}(t_i))|e^{\lambda(t_i - \tau_{ij}(t_i))} e^{\lambda \tau_{ij}(t_i)} + \sum_{j=1}^{n}\sum_{l=1}^{n} \overline{a_i}|e_{ijl}(t_i)|(L_j^h M_l^h |y_j(t_i - \sigma_{ijl}(t_i))| \\
&\quad \times e^{\lambda(t_i - \sigma_{ijl}(t_i))} e^{\lambda \sigma_{ijl}(t_i)} + M_j^h L_l^h |y_l(t_i - v_{ijl}(t_i))|e^{\lambda(t_i - v_{ijl}(t_i))} e^{\lambda v_{ijl}(t_i)}\Bigg] e^{\lambda t_i} \\
&\le [\lambda - L_i^{ab} + \sum_{j=1}^{n} |c_{ij}(t_i)|\overline{a_i}L_j^f + \sum_{j=1}^{n} |c_{ij}(t_i)|L_j^a L_j^f \gamma + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g \gamma \\
&\quad + \sum_{j=1}^{n}\sum_{l=1}^{n} |e_{ijl}(t_i)|L_j^a M_j^h M_l^h + L_i^a I_i(t_i)]M||\varphi - \varphi^*||_\infty + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g e^{\lambda\tau} \\
&\quad M||\varphi - \varphi^*||_\infty + \sum_{j=1}^{n}\sum_{l=1}^{n} \overline{a_i}|e_{ijl}(t_i)|(L_j^h M_l^h + M_j^h L_l^h)e^{\lambda\tau} M||\varphi - \varphi^*||_\infty \\
&\le \Bigg[\lambda - L_i^{ab} + \sum_{j=1}^{n} |c_{ij}(t_i)|\overline{a_i}L_j^f + \sum_{j=1}^{n} |c_{ij}(t_i)|L_j^a L_j^f \gamma + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g \gamma \\
&\quad + \sum_{j=1}^{n}\sum_{l=1}^{n} |e_{ijl}(t_i)|L_j^a M_j^h M_l^h + L_i^a I_i(t_i) + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g e^{\lambda\tau} \\
&\quad + \sum_{j=1}^{n}\sum_{l=1}^{n} \overline{a_i}|e_{ijl}(t_i)|(L_j^h M_l^h + M_j^h L_l^h)e^{\lambda\tau}\Bigg] M||\varphi - \varphi^*||_\infty. \qquad (17)
\end{aligned}$$

Thus,

$$0 \le \lambda - L_i^{ab} + \sum_{j=1}^{n} |c_{ij}(t_i)|\overline{a_i}L_j^f + \sum_{j=1}^{n} |c_{ij}(t_i)|L_j^a L_j^f \gamma + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g \gamma$$
$$+ \sum_{j=1}^{n}\sum_{l=1}^{n} |e_{ijl}(t_i)|L_j^a M_j^h M_l^h + L_i^a I_i(t_i) + \sum_{j=1}^{n} |d_{ij}(t_i)|\overline{a_i}L_j^g e^{\lambda\tau}$$
$$+ \sum_{j=1}^{n}\sum_{l=1}^{n} \overline{a_i}|e_{ijl}(t_i)|(L_j^h M_l^h + M_j^h L_l^h)e^{\lambda\tau} \triangleq H.$$

Which contradicts (H4), hence (14) holds. It follows that $|x_i(t) - x_i^*(t)| = |y_i(t)| < M\|\varphi - \varphi^*\|_\infty e^{-\lambda t}, i = 1, 2, \ldots, n,\ t > 0$. This completes the proof of Lemma 2.

Remark 2. If $x^*(t) = (x_1^*(t), x_2^*(t), \ldots, x_n^*(t))^T$ is the T-anti-periodic solution of system (1) with initial value φ^* being bounded by γ, it follows from Lemma 2 and Definition 1 that $x^*(t)$ is globally exponentially stable.

3 Main Results

The following is our main result.

Theorem 1. *Suppose that $(H1) - (H4)$ are satisfied. Then system (1) has exactly one T-anti-periodic solution $x^*(t)$. Moreover, $x^*(t)$ is globally exponentially stable.*

Proof. Let $v(t) = (v_1(t), v_2(t), \ldots, v_n(t))^T$ be a solution of system(1) with initial conditions

$$v_i(s) = \varphi_i^\nu(s), \quad |\varphi_i^\nu(s)| < \gamma, \quad s \in (-\infty, 0],\ \ i = 1, 2, \ldots, n. \tag{18}$$

According to Remark 1, $v(t)$ exists on [0,∞). Moreover, by Lemma 1, the solution $v(t)$ is bounded and

$$|v_i(t)| < \gamma, \quad \text{for all } t \in [-\tau, \infty),\ \ i = 1, 2, \ldots, n. \tag{19}$$

Form (2)-(5), we have

$$((-1)^{k+1}v_i(t + (k+1)T))' = (-1)^{k+1}v_i'(t + (k+1)T)$$
$$= (-1)^{k+1}\{-a_i(v_i(t + (k+1)T))[b_i(v_i(t + (k+1)T)) - \sum_{j=1}^{n} c_{ij}(t + (k+1)T)$$
$$\times f_j(v_j(t + (k+1)T)) - \sum_{j=1}^{n} d_{ij}(t + (k+1)T)g_j(v_j(t + (k+1)T - \tau_{ij}(t$$
$$+(k+1)T))) - \sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t + (k+1)T)h_j(v_j(t + (k+1)T - \sigma_{ijl}(t$$

$$
\begin{aligned}
&+(k+1))T))h_l(v_l(t+(k+1)T-\nu_{ijl}(t+(k+1)T)))-I_i(t+(k+1)T)]\}\\
&=(-1)^{k+1}\{-a_i(v_i(t+(k+1)T))[b_i(v_i(t+(k+1)T))-\sum_{j=1}^{n}c_{ij}(t+(k+1)T)\\
&\times f_j(v_j(t+(k+1)T))-\sum_{j=1}^{n}d_{ij}(t+(k+1)T)g_j(v_j(t+(k+1)T-\tau_{ij}(t)))\\
&-\sum_{j=1}^{n}\sum_{l=1}^{n}e_{ijl}(t+(k+1)T)h_j(v_j(t+(k+1)T-\sigma_{ijl}(t))h_l(v_l(t+(k+1)T\\
&-\nu_{ijl}(t)))-I_i(t+(k+1)T)]\}\\
&=-a_i((-1)^{k+1}v_i(t+(k+1)T))[b_i((-1)^{k+1}v_i(t+(k+1)T))-\sum_{j=1}^{n}c_{ij}(t)\\
&\times f_j((-1)^{k+1}v_j(t+(k+1)T))-\sum_{j=1}^{n}d_{ij}(t+(k+1)T)g_j(v_j(t+(k+1)T\\
&-\tau_{ij}(t)))-\sum_{j=1}^{n}\sum_{l=1}^{n}e_{ijl}(t)h_j((-1)^{k+1}v_j(t+(k+1)T)-\sigma_{ijl}(t))h_l((-1)^{k+1}\\
&\times v_l(t+(k+1)T-\nu_{ijl}(t)))-I_i(t)],i=1,2,\ldots,n. \qquad (20)
\end{aligned}
$$

Thus, for any natural number $k,(-1)^{k+1}v(t+(k+1)T)$ are the solution of system(1) on R. Then, by Lemma 2, there exists a constant $M>0$ such that

$$
\begin{aligned}
&|(-1)^{k+1}(v_i(t+(k+1)T)-(-1)^k v_i(t+kT)|\\
&\le Me^{-\lambda(t+kT)}\sup_{-\tau\le s\le 0}\max_{1\le i\le n}|v_i(s+T)+v_i(s)|\\
&\le 2e^{-\lambda(t+kT)}M\gamma,\ \forall\ t+kT>0,i=1,2,\ldots,n. \qquad (21)
\end{aligned}
$$

Thus, for any natural number m, we obtain

$$
\begin{aligned}
(-1)^{m+1}v_i(t+(m+1)T)=v_i(t)+\sum_{k=0}^{m}[&(-1)^{k+1}v_i(t+(k+1)T)-\\
&(-1)^k v_i(t+kT)]. \qquad (22)
\end{aligned}
$$

Then,

$$
\begin{aligned}
|(-1)^{m+1}v_i(t+(m+1)T)|\le |v_i(t)|+\sum_{k=0}^{m}&|(-1)^{k+1}v_i(t+(k+1)T)\\
&-(-1)^m v_i(t+kT)|,\ i=1,2,\ldots,n. \qquad (23)
\end{aligned}
$$

In view of (21), we can choose a sufficiently large constant $P>0$ and a positive constant $\alpha>0$, for $i=1,2,\ldots,n$ such that

$$
|(-1)^{k+1}v_i(t+(k+1)T)-(-1)^k v_i(t+kT)|\le \alpha e^{-\lambda\tau k},\forall\ k>P, \qquad (24)
$$

on any compact set of R. It follows from (22)-(24) that $(-1)^m v(t+mT)$ uniformly converges to a continuous function $x^*(t) = (x_1^*(t), x_2^*(t), \ldots, x_n^*(t))^T$ on any compact set of R.

Now we will show that $x^*(t)$ is T-anti-periodic solution of system (1). First, $x^*(t)$ is T-anti-periodic, since,

$$\begin{aligned} x^*(t+T) &= \lim_{m\to\infty} (-1)^m v(t+T+mT) \\ &= -\lim_{(m+1)\to\infty} (-1)^{m+1} v(t+(m+1)T) = -x^*(t). \end{aligned}$$

Next, we prove that $x^*(t)$ is a solution of system(1). In fact, together with the continuity of the right side of system(1) and (20) implies that $\{((-1)^{m+1}v(t+(m+1)T))'\}$ uniformly converges to a continuous function on any compact set of R. Thus, letting $m \to \infty$, we obtain

$$\begin{aligned} \frac{d\{x_i^*(t)\}}{\mathrm{d}t} = -a_i(x_i^*(t))\Big[& b_i(x_i^*(t)) - \sum_{j=1}^{n} c_{ij}(t) f_j(x_j^*(t)) - \sum_{j=1}^{n} d_{ij}(t) g_j(x_j^*(t \\ & -\tau_{ij}(t))) - \sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t) h_j(x_j^*(t-\sigma_{ijl}(t))) h_l(x_l^*(t-\nu_{ijl}(t))) - I_i(t)\Big] \end{aligned}$$

Therefore, $x_i^*(t)$ is a solution of system (1).

Finally, from Remark 1 and by Lemma 2, we know that (1) has an T-anti-periodic solution, and we can prove that $x^*(t)$ is exponentially stable. This completes the proof.

4 An Example

Let $n = 2$, consider the following high-order Cohen-Grossberg neural networks with delays:

$$\begin{aligned} x_i'(t) = -a_i(x_i(t))\Big[& b_i(x_i(t)) - \sum_{j=1}^{2} c_{ij}(t) f_j(x_j(t)) - \sum_{j=1}^{2} d_{ij}(t) g_j(x_j(t-\tau_{ij}(t))) \\ & - \sum_{j=1}^{2}\sum_{l=1}^{2} e_{ijl}(t) h_j(x_j(t-\sigma_{ijl}(t))) h_l(x_l(t-\nu_{ijl}(t))) - I_i(t)\Big], \end{aligned} \tag{25}$$

where

$$\begin{aligned} & a_1(x_1(t)) = 3 + |\cos(x_1(t))|, \quad a_2(x_2(t)) = 3 + |\sin(x_2(t))|, \\ & b_1(x_1(t)) = \frac{8x_1(t) + \sin(x_1(t))}{2 + |\cos(x_1(t))|}, \quad b_2(x_2(t)) = \frac{8x_2(t) + \sin(x_2(t))}{2 + |\cos(x_2(t))|}, \\ & c_{ij}(t) = \begin{pmatrix} 1/16|\sin t| & 1/32\|\cos t| \\ 1/6|\cos t| & 1/15|\sin t| \end{pmatrix}, \; d_{ij}(t) = 1/16, \; e_{112} = e_{212} = \frac{1}{8}(\sin t), \\ & ijl \neq 212, ijl \neq 112 \;\; , \;\; \sigma_{112}(t) = 5\sin^2 t, \;\; \sigma_{212}(t) = \sin^2 t, \nu_{112} = 2\sin^2 t, \\ & \nu_{212} = 4\sin^2 t, \;\; \tau_{ij}(t) = 1/2\cos^2(t), f_j(u) = \sin u, \;\; g_j(u) = 1/2\sin u, \\ & h_i(t) = 1/2(|x+1| - |x-1|), \;\; I_1(t) = 2\sin t, \;\; I_2(t) = \sin t. \end{aligned}$$

Note that
$L_1^{ab} = L_2^{ab} = 7$, $\overline{a_1} = \overline{a_2} = 4$, $L_1^f = L_2^f = 1$, $L_1^h = L_2^h = 1/2$, $L_1^g = L_2^h = M_1^g = M_2^h = 1$. Then

$$\max_{1\le i\le 2}\Bigg\{(L_i^{ab} - \sum_{j=1}^{2}|c_{ij}(t_i)|\overline{a_i}L_j^f - \sum_{j=1}^{n}|c_{ij}(t_i)|L_j^aL_j^f\gamma - \sum_{j=1}^{n}|d_{ij}(t_i)|\overline{a_i}L_j^g\gamma$$
$$-\sum_{j=1}^{2}\sum_{l=1}^{2}|e_{ijl}(t_i)|L_j^aM_j^hM_l^h - L_i^aI_i(t_i))^{-1}\Bigg(\sum_{j=1}^{n}|d_{ij}(t_i)|\overline{a_i}L_j^g$$
$$+\sum_{j=1}^{2}\sum_{l=1}^{n}\overline{a_i}|e_{ijl}(t_i)|(L_j^hM_l^h + M_j^hL_l^h)\Bigg)\Bigg\} < 1.$$

where $i = 1, 2$, which implies that system (25) satisfies all the conditions in Theorem 1. Hence, system (25) has exactly one T-anti-periodic solution. Moreover, the T-anti-periodic solution if globally exponentially stable.

References

1. Cohen, M., Grossberg, S.: IEEE Trans. Syst. Man. Cybern. 13, 815 (1983)
2. Li, Y.: Criteria for exponential stability of Cohen-Grossberg neural networks. Neural Networks 17, 1401–1414 (2004)
3. Li, Y.: Global stability and existence of periodic solutions of discrete delayed cellular neural networks. Physics Letters A 333, 51–61 (2004)
4. Li, Y.: Existence stability of periodic solutions for Cohen-Grossberg neural networks with multiple delays. Chaos Solitons Fract. 20, 459–466 (2004)
5. Su, W., Chen, Y.: Global robust stability criteria of stochastic Cohen-Grossberg neural networks with discrete and distributed time-varying delays. Commun. Nonlinear Sci. Numer. Simul. 14, 520–528 (2009)
6. Ou, C.: Anti-periodic solutions for high-order Hopfield neural networks. Comput. Math. Appl. 56, 1838–1844 (2008)
7. Xiao, B., Meng, H.: Existence and exponential stability of positive almost periodic solutions for high-order Hopfield neural networks. Appl. Math. Modell. 39, 2133–2137 (2009)
8. Zhang, F., Li, Y.: Almost periodic solutions for higher-order Hopfield neural networks without bounded activation functions. Electron. J. Diff. Eqns. 97, 1–10 (2007)
9. Aftabizadeh, A.R., Aizicovici, S., Pavel, N.H.: On a class of second-order anti-periodic boundary value problems. J. Math. Anal. and Appl. 171, 301–320 (1992)
10. Johnson, C.R., Smith, R.L.: The completion problem for M-matrices and inverse M-matrices. Linear Algebra and Its Applications, 241–243 (1996)
11. Liu, X., Cao, J.: Exponential stability of anti-periodic solutions for neural networks with multiple discrete and distributed delays. IMechE, Part I: J. Systems and Control Engineering 223, 299–308 (2009)
12. Hale, J.K.: Theory of Functional Differential Equations. Springer, New York (1977)

Global Exponential Stability of BAM Type Cohen-Grossberg Neural Network with Delays on Time Scales*

Chaolong Zhang[1], Wensi Ding[2,**], Fengjian Yang[1], and Wei Li[1]

[1] Department of Computation Science, Zhongkai University of Agriculture and Engineering, Guangzhou, 510225, P.R. China
[2] School of Machanical and Automative Engineering of South China University of Technology, Guangzhou, 510640, P.R. China
zhcl88@126.com

Abstract. In this paper,a class of the BAM type Cohen-Grossberg neural network with delays is considered on time scales. Some sufficient conditions about the existence of equilibrium point, globally exponential stability and globally exponentially robust stability are obtained.At last,an example can be demonstrate the results.

Keywords: Cohen-Grossberg BAM neural network, Lyapunov function, time scales.

1 Introduction

Cohen-Grossberg BAM neural networks aroused extensive researches by numerous scholars due to its applicability in solving image processing, signal processing and pattern recognition problems.The dynamical behaviors of neural network has been extensively studied in the literature, e.g., Ref[1-10].

In Ref.[11], on the basis of the structure of BAMs neural network, the authors obtained some sufficient conditions of globally exponential stability of solutions.

In this paper,we consider a general class of Cohen-Grossberg neural networks with delay on time scales:

$$\begin{cases} x_i^{\Delta}(t)=-a_i(x_i(t))[b_i(x_i(t))-\sum_{j=1}^{m} p_{ji}(t)f_j(y_j(t))-\sum_{j=1}^{m} \overline{p}_{ji}(t)f_j(y_j(t-\tau_{ji}))+I_i(t)], & t\in \mathrm{T}, t>0, \\ y_j^{\Delta}(t)=-a_j(y_j(t))[b_j(y_j(t))-\sum_{i=1}^{n} q_{ij}(t)g_i(x_i(t))-\sum_{i=1}^{n} \overline{q}_{ij}(t)g_i(x_i(t-\sigma_{ij}))+J_j(t)], & t\in \mathrm{T}, t>0, \end{cases} \quad (1)$$

where the functions a_i, a_j represent the abstract amplification function, while the function b_i, b_j represent the self-excitation rate function, $p_{ij}, q_{ji}, \overline{p}_{ij}, \overline{q}_{ji,}$ are the connection weights. $I_i(t), J_j(t)$ denote the ith and jth component of an external input source introduced from outside the network to the cell the cell i and j, respectively.

* This project was supported by the NSF of Guangdong Province of China (No.8151027501000053) and National Natural Science Foundation (50775075) and Project of Zhongkai University Agriculture and Engineering(G3071728).

** Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 595–602, 2010.

The system (1) is supplemented with initial values given by

$$\begin{cases} x_i(s) = \varphi_i(s), s \in [-\tau, 0]_T, \tau = \max\limits_{1\le i\le n, 1\le j\le m} \{\tau_{ji}\}, i = 1, 2, \cdots, n \\ y_j(s) = \psi_j(s), s \in [-\sigma, 0]_T, \sigma = \max\limits_{1\le i\le n, 1\le j\le m} \{\sigma_{ij}\}, j = 1, 2, \cdots, m, \end{cases}$$

where $\varphi_i \in C([-\tau, 0]_T, R), \psi_j \in ([-\sigma, 0]_T, R)$.

2 Preliminaries

To begin with,we introduce some notation and recall some basic definitions.

A time scale T is an arbitrary non-empty closed subset of the real numbers R.On any time scale we define theforward and backward jump operators by

$$\sigma(t) := \inf\{s \in T : s > t\}, \quad \rho(t) := \sup\{s \in T : s < t\} \tag{2}$$

A point t is said to be left-dense if $\rho(t) = t$, right-dense if $\sigma(t) = t$, left scattered if $\rho(t) < t$, and right scattered if $\sigma(t) > t$.The graininess μ of the time scale is define by $\mu(t) = \sigma(t) - t$. The set T^k is derived from T as follow: if T has a left-scattered maximum m, then $T^k = T - \{m\}$; otherwise, $T^k = T$.

For $f: T \to R$ and $t \in T^k$, we define $f^{\Delta}(t)$ to be the number (provided it exists) such that given any $\varepsilon > 0$, there is a neighborhood U of t with

$$\left|[f(\sigma(t)) - f(s)] - f^{\Delta}(t)[\sigma(t) - s]\right| \le \varepsilon \left|\sigma(t) - s.\right|$$

We call $f^{\Delta}(t)$ the delta derivative of f at t.

A function $f: T \to R$ is said to be rd-continuous at each right-dense point and if there exists a finite left limit in all left-dense points, and f is said to be differentiable if its derivative exists.The set of rd-continuous functions $f: T \to R$ will be denoted by $C_{rd}(T, R)$.

Lemma 1. (Bohnerand Peterson [12]).Assume $f, g: T \to R$ are differential at $t \in T^k$, then

(i) $(fg)^{\Delta} = f^{\Delta}(t)g(t) + f(\sigma(t))g^{\Delta}(t) = f(t)g^{\Delta}(t) + f^{\Delta}(t)g(\sigma(t))$

(ii) $(\frac{f}{g})^{\Delta}(t) = \frac{f^{\Delta}(t)g(t) - f(t)g^{\Delta}(t)}{g(t)g(\sigma(t))}$.

Lemma 2. (Bohner and Peterson [12]).If $a, b, c \in T, \alpha, \beta \in R$ and $f, g: T \to R$ then

(i) $\int_a^b f^{\Delta}(t)\Delta t = f(b) - f(a)$;

(ii) $\int_a^b [\alpha f(t) + \beta g(t)]\Delta t = \alpha \int_a^b f(t)\Delta t + \beta \int_a^b g(t)\Delta t$;

(iii) $\int_a^b f(t)\Delta t = \int_a^c f(t)\Delta t + \int_c^b f(t)\Delta t;$

(iv) $\int_a^b f(t)g^{\Delta}(t)\Delta t = (fg)(b) - (fg)(a) - \int_a^b f^{\Delta}(t)g(t)\Delta t;$

(v) if $|f(t)| \le g(t)$ on $[a,b)$, then $\left|\int_a^b f(t)\Delta t\right| \le \int_a^b g(t)\Delta t;$

(vi) if $f(t) \ge 0$ for all $a \le t \le b$ then $\int_a^b f(t)\Delta t \ge 0.$

A function p is a regressive function,then the generalized exponential function e_p is defined by

$$e_p(t,s) = \exp\left(\int_s^t \xi_{\mu(t)}(p(\tau))\Delta\tau\right), \quad \text{for} \quad s,t \in T \tag{3}$$

with the cylinder transformation $\xi_h(z) = \begin{cases} \dfrac{Log(1+hz)}{h}, & h \neq 0, \\ z, & h = 0. \end{cases}$

Lemma 3. (Bohner and Peterson[12]). If $p,q \in \Re$ then

(i) $e_p(t,t) \equiv 1$ and $e_0(t,s) \equiv 1;$

(ii) $e_p(\sigma(t),s) = (1+\mu(t)p(t))e_p(t,s);$

(iii) $e_p(t,s) = \dfrac{1}{e_p(s,t)};$

(iv) $e_p(t,s)e_p(s,r) = e_p(t,r);$

(v) $e_p(t,s)e_q(t,s) = e_{p\oplus q}(t,s);$

(vi) $e_p(t,t_0) > 0$ for all $p > 0, t \in T.$

Definition 1. (Lakshmikantham and Vatsala [13]) For each $t \in T$, let N be a neighborhood of t. Then $V \in C_{rd}[T \times R^n, R^+]$, define $D^+V^{\Delta}(t,x(t))$ to mean that. given $\varepsilon > 0$, there exists a right neighborhood $N_\varepsilon \subset N$ of t such that

$$\frac{1}{\mu(t,s)}[V(\sigma(t),x(\sigma(t)) - V(s,x(\sigma(t))) - \mu(t,s)f(t,x(t))] < D^+V^{\Delta}(t,x(t)) + \varepsilon$$

for each $s \in N_\varepsilon, s > t$ where $\mu(t,s) = \sigma(t) - t$. If t is rs and $V(t,x(t))$ is continuous at t, this is reduces to

$$D^+V^{\Delta}(t,x(t)) = \frac{V(\sigma(t),x(\sigma(t))) - V(t,x(\sigma(t))}{\sigma(t) - t}$$

Definition 2. The solution $u^*(t) = (x_1^*(t),\cdots,x_n^*(t), y_1^*(t),\cdots,y_m^*(t))^T$ of system (1) is said to be exponentially stable if there exist a positive constant α such that for every $\delta \in T$, there exists $N = N(\delta) \ge 1$ such that the solution $u(t) = (x_1(t),\cdots, x_n(t), y_1(t),\cdots,y_m(t))^T$ satisfies

$$\left\|u(t)-u^*(t)\right\| \le N\frac{1}{e_\alpha(t,\delta)}\left(\sum_{i=1}^{n}\left|\varphi_i(\delta)-x_i^*(\delta)\right|+\sum_{j=1}^{m}\left|\psi_j(\delta)-y_j^*(\delta)\right|\right), t\in T^+$$

where $\delta\in[-\max\{\tau,\sigma\},0]_T$.

Throughout this paper, we assume that

(H_1) $a_l(\bullet)>0, a_l$ are bounded, that is, there exist $\underline{a}_l, \overline{a}_l>0$ such that $\underline{a}_l \le a \le \overline{a}_l$, $l=1,2,\cdots,\max\{m,n\}$, $\underline{a}=\min\{\underline{a}_l\}$, $\overline{a}=\max\{\overline{a}_l\}$;

(H_2) $0\le \underline{\gamma}_l \le b_l^{\Delta}(\bullet)\le \overline{\gamma}_l$ for any $l=1,2,\cdots,\max\{m,n\}$;

(H_3) there exist $L_j^f>0, L_i^g>0$ such that

$$\left|f_j(u)-f_j(v)\right|\le L_j^f\left|u-v\right|, \left|g_i(u)-g_i(v)\right|\le L_i^g\left|u-v\right|,$$

for any $u,v\in R, i=1,2,\cdots,n; j=1,2,\cdots,m.$

In the following,we use the following norm of R^{n+m}:

$$\left\|w\right\|=\sum_{l=1}^{n+m}\left|w_l\right|, w=(w_1,w_2,\cdots,w_{n+m})^T\in R^{n+m}$$

Assume that $(x_1^*,\cdots,x_n^*,y_1^*,\cdots,y_m^*)^T$ is an unique equilibrium point of system (1). Let

$$z_i(t)=x_i(t)-x_i^*, \overline{z}_j(t)=y_j(t)-y_j^*, \alpha_i(z_i(t))=a_i(z_i(t)+x_i^*),$$

$$\alpha_j(\overline{z}_j(t))=a_j(\overline{z}_j(t)+y_j^*), \beta_i(z_i(t))=b_i(z_i(t)+x_i^*)-b_i(x_i^*),$$

$$\beta_j(\overline{z}_j(t))=b_j(\overline{z}_j(t)+y_j^*)-b_j(y_j^*), \psi_i(z_i(t))=g_i(z_i(t)+x_i^*)-g_i(x_i^*),$$

$$\varphi_j(\overline{z}_j(t))=f_j(\overline{z}_j(t)+y_j^*)-f_j(y_j^*), z(t)=(z_1(t),\cdots,z_n(t),\overline{z}_1(t),\cdots,\overline{z}_m(t))^T,$$

then the system (1) can be written in the following form:

$$\begin{cases} z_i^{\Delta}(t)=-\alpha_i(z_i(t))[\beta_i(z_i(t))-\sum_{j=1}^{m}p_{ji}(t)\varphi_j(\overline{z}_j(t))-\sum_{j=1}^{m}\overline{p}_{ji}(t)\varphi_j(\overline{z}_j(t-\tau_{ji}))], \\ \overline{z}_j^{\Delta}(t)=-\alpha_j(\overline{z}_j(t))[\beta_j(\overline{z}_j(t))-\sum_{i=1}^{n}q_{ij}(t)\psi_i(z_i(t))-\sum_{i=1}^{n}\overline{q}_{ij}(t)\psi_i(z_i(t-\sigma_{ij}))]. \end{cases} \tag{4}$$

3 Main Results

Theorem 1. Assume that

$$\ker(f,g)=\left\{(x,y)\in R^2\middle|f_j(y)=0,g_i(x)=0\right\}\cap\left\{(x_i,y_j)\middle|b_i(x_i(t))=-I_i(t),b_j(y_j(t))=-J_j(t)\right\}\neq\varnothing,$$

then the system (1) has at least an equilibrium point.

Proof. If $(x_1^*,\cdots,x_n^*,y_1^*,\cdots,y_m^*)^T$ is an equilibrium point of system (1), then we have

$$\begin{cases} b_i(x_i^*) = \sum_{j=1}^{m} (p_{ji} + \overline{p}_{ji}) f_j(y_j^*) - I_i(t) \\ b_j(y_j^*) = \sum_{i=1}^{n} (q_{ij} + \overline{q}_{ij}) g_i(x_i^*) - J_j(t) \end{cases} \tag{5}$$

which is equivalent to

$$\begin{cases} b_i(x_i^*) = \sum_{j=1}^{m} f_j(y_j^*) - I_i(t) \\ b_j(y_j^*) = \sum_{i=1}^{n} g_i(x_i^*) - J_j(t) \\ \sum_{j=1}^{m} (p_{ji} + \overline{p}_{ji} - 1) f_j(y_j^*) = 0 \\ \sum_{i=1}^{n} (q_{ij} + \overline{q}_{ij} - 1) g_i(x_i^*) = 0, \end{cases} \tag{6}$$

form the third and forth equations of (6),we can see that any element in $\ker(f,g)$ is solution. So, by assumption,we can know that there exist some element in $\ker(f,g)$ which is solution of the first and second equations of (6). So the system (1) has at least an equilibrium point.

Corollary 1. Assume that the conditions of Theorem 1, as well as the hypothesis (H_1)–(H_3) hold, then the system (1) has an unique equilibrium point.

Lemma 4. (Halanay inequality) Assume that there exists $k_1 > k_2 > 0$, $h(t) \in C[t_0-\tau, t_0]_T$ is non-negative, $\tau > 0, h^\tau(t) = \sup_{s\in[t-\tau,t]_T}\{h(s)\}$, and

$$D^+h^\Delta(t) \le -k_1 h(t) + k_2 h^\tau(t),$$

then $h(t) \le h^\tau(t_0)\dfrac{1}{e_\lambda(t,t_0)}$, $t \ge t_0$, where λ is unique solution of the equation $\lambda = k_1 - k_2 e_\lambda(0,-\tau)$.

Theorem 2. Under the conditions in Theorem 1, Assume that (H_1)–(H_3) hold, and

(i) $|p_{ji}(t)| \le p_{ji}, |q_{ij}(t)| \le q_{ij}, |\overline{p}_{ji}(t)| \le \overline{p}_{ji}, |\overline{q}_{ij}(t)| \le \overline{q}_{ij}$,

where $p_{ji}, q_{ij}, \overline{p}_{ji}, \overline{q}_{ij}$ are positive constants;

(ii) $k_1 = \underline{a} \min_{1\le i\le n.1\le j\le m}\left\{\underline{\gamma}_i - \sum_{j=1}^{m}|q_{ij}|L_i^g, \underline{\gamma}_j - \sum_{i=1}^{n}|p_{ji}|L_j^f,\right\} > k_2 = \overline{a} \max_{1\le i\le n.1\le j\le m}\left\{\sum_{j=1}^{m}|\overline{q}_{ij}|L_i^g, \sum_{i=1}^{n}|\overline{p}_{ji}|L_j^f,\right\} > 0;$

then the equilibrium point of system (1) is globally exponentially stable.

Proof. Consider Lyapunov function as follows:

$$V(t)=\sum_{i=1}^{n}\int_{0}^{z_i(t)}\frac{Sgns}{\alpha_i(s)}\Delta s+\sum_{j=1}^{m}\int_{0}^{\bar{z}_j(t)}\frac{Sgns}{\alpha_j(s)}\Delta s,$$

then

$$\frac{1}{\bar{a}}\|z(t)\|\le V(t)\le\frac{1}{\underline{a}}\|z(t)\|, \tag{7}$$

Since $\sum_{j=1}^{m}\sum_{i=1}^{n}\eta_{ij}\mu_i=\sum_{i=1}^{n}\sum_{j=1}^{m}\eta_{ij}\mu_i$, we have

$$\begin{aligned}
V^{\Delta}(t)&=-\sum_{i=1}^{n}Sgnz_i(t)[\beta_i(z_i(t))-\sum_{j=1}^{m}p_{ji}(t)\varphi_j(\bar{z}_j(t))-\sum_{j=1}^{m}\bar{p}_{ji}(t)\varphi_j(\bar{z}_j(t-\tau_{ji}))]\\
&\quad-\sum_{j=1}^{m}Sgn\bar{z}_j(t)[\beta_j(\bar{z}_j(t))-\sum_{i=1}^{n}q_{ij}(t)\psi_i(z_i(t))-\sum_{i=1}^{n}\bar{q}_{ij}(t)\psi_i(z_i(t-\sigma_{ij}))]\\
&\le-\sum_{i=1}^{n}[\underline{\gamma}_i|z_i(t)|-\sum_{j=1}^{m}p_{ji}L_j^f|\bar{z}_j(t)|-\sum_{j=1}^{m}\bar{p}_{ji}L_j^f|\bar{z}_j(t-\tau_{ji})|]\\
&\quad-\sum_{j=1}^{m}[\underline{\gamma}_j|\bar{z}_j(t)|-\sum_{i=1}^{n}q_{ij}L_i^g|z_i(t)|-\sum_{i=1}^{n}\bar{q}_{ij}L_i^g|z_i(t-\sigma_{ij})|]\\
&=-\sum_{i=1}^{n}\left(\underline{\gamma}_i-\sum_{j=1}^{m}q_{ij}L_i^g\right)|z_i(t)|+\sum_{i=1}^{n}\sum_{j=1}^{m}\bar{q}_{ij}L_i^g|z_i(t-\sigma_{ij})|\\
&\quad-\sum_{j=1}^{m}\left(\underline{\gamma}_j-\sum_{i=1}^{n}p_{ji}L_j^f\right)|\bar{z}_j(t)|+\sum_{i=1}^{n}\sum_{j=1}^{m}\bar{p}_{ji}L_j^f|\bar{z}_j(t-\tau_{ji})|\\
&\le-\frac{k_1}{\underline{a}}\|z(t)\|+\frac{k_2}{\bar{a}}\|\bar{z}^{\tau}(t)\|\le-k_1V(t)+k_2V^{\tau}(t).
\end{aligned}$$

Since $k_1>k_2$, by Halanay inequality, it follows that

$$V(t)\le V^{\tau}(t_0)\frac{1}{e_\lambda(t,t_0)}, \tag{8}$$

by (8), we have $$V(t)\le N\frac{1}{e_\lambda(t,t_0)}. \tag{9}$$

where $N=V^{\tau}(t_0)=\sup_{s\in[t_0-\tau,t_0]_T}\{V(s)\}$, from the Eq.(7), we have

$$\|z(t)\|\le\bar{a}N\frac{1}{e_\lambda(t,t_0)},\ t>0.$$

i.e., $|x_i(t)-x_i^*|+|y_j(t)-y_j^*|\le\bar{a}N\frac{1}{e_\lambda(t,t_0)},\ t>0.$

Therefore, we know the equilibrium point of system (1) is globally exponentially stable.

In this section, we assume that

$$\underline{a}_{ji} \le p_{ji}(t) \le \overline{a}_{ji}, \underline{b}_{ij} \le q_{ij}(t) \le \overline{b}_{ij}, \underline{c}_{ji} \le \overline{p}_{ji}(t) \le \overline{c}_{ji}, \underline{d}_{ij} \le \overline{q}_{ij}(t) \le \overline{d}_{ij},$$

where $\underline{a}_{ji}, \overline{a}_{ji}, \underline{b}_{ij}, \overline{b}_{ij}, \underline{c}_{ji}, \overline{c}_{ji}, \underline{d}_{ij}, \overline{d}_{ij}$ are constant. Let

$$p_{ji}^{(0)} = \frac{1}{2}(\overline{a}_{ji} + \underline{a}_{ji}), p_{ji}^{(1)} = \frac{1}{2}(\overline{a}_{ji} - \underline{a}_{ji}), q_{ij}^{(0)} = \frac{1}{2}(\overline{b}_{ij} + \underline{b}_{ij}), q_{ij}^{(1)} = \frac{1}{2}(\overline{b}_{ij} - \underline{b}_{ij}),$$

$$\overline{p}_{ji}^{(0)} = \frac{1}{2}(\overline{c}_{ji} + \underline{c}_{ji}), \overline{p}_{ji}^{(1)} = \frac{1}{2}(\overline{c}_{ji} - \underline{c}_{ji}), \overline{q}_{ij}^{(0)} = \frac{1}{2}(\overline{d}_{ij} + \underline{d}_{ij}), \overline{q}_{ij}^{(1)} = \frac{1}{2}(\overline{d}_{ij} - \underline{d}_{ij}),$$

$$k_1 = \underline{a} \min_{1\le i\le n, 1\le j\le m} \left\{ \gamma_i - \sum_{j=1}^{m} (|q_{ij}^{(0)}| + q_{ij}^{(1)}) L_i^g, \gamma_j - \sum_{i=1}^{n} (|p_{ji}^{(0)}| + p_{ji}^{(1)}) L_j^f, \right\},$$

$$k_2 = \overline{a} \max_{1\le i\le n, 1\le j\le m} \left\{ \sum_{j=1}^{m} (|\overline{q}_{ij}^{(0)}| + \overline{q}_{ij}^{(1)}) L_i^g, \sum_{i=1}^{n} (|\overline{p}_{ji}^{(0)}| + \overline{p}_{ji}^{(1)}) L_j^f, \right\},$$

Theorem 3. Under the conditions in the Theorem 1, assumethat (H_1)–(H_3) hold, and $k_1 > k_2 > 0$, then the equilibrium point of system (1) is globally robust exponentially stable.

Proof. Note $|p_{ji}(t)| \le |p_{ji}^{(0)}| + p_{ji}^{(1)}, |q_{ij}(t)| \le |q_{ij}^{(0)}| + q_{ij}^{(1)}, |\overline{p}_{ji}(t)| \le |\overline{p}_{ji}^{(0)}| + \overline{p}_{ji}^{(1)}, |\overline{q}_{ij}(t)| \le |\overline{q}_{ij}^{(0)}| + \overline{q}_{ij}^{(1)}$, similar to the proof of theorem 2, it is easy to prove the above theorem.

4 Example

Consider the following system:

$$\begin{cases} x_1^{\Delta}(t) = -(6 - \cos t)[x_1(t) - \frac{1}{10}\sin t \cos(y_1(t)) - \frac{1}{21}\sin t \cos(y_1(t-\tau)) - \pi], \\ y_1^{\Delta}(t) = -(5 + \sin t)[y_1(t) - \frac{1}{12}\cos t \sin(x_1(t)) - \frac{1}{18}\cos t \sin(x_1(t-\sigma)) - \frac{\pi}{2}], \end{cases} \tag{10}$$

where $\tau = \sigma = 1, I_i(t) = -\pi, I_j(t) = -\frac{\pi}{2}$. We can compute $\underline{a} = 4, \overline{a} = 7, p_{11} = \frac{1}{10}, q_{11} = \frac{1}{12}$, $\overline{p}_{11} = \frac{1}{21}, \overline{q}_{11} = \frac{1}{18}, k_1 = \frac{9}{10}, k_2 = \frac{1}{5}, L_1^f = L_1^g = 1$.

From the above assumption,the conditions of Theorem 2 are satisfied. Therefore, the equilibrium point $(\pi, \frac{\pi}{2})$ of system (10) is globally exponentially stable.

References

1. Belair, J.: Stability in a model of a delayed neural network. J. Dynam. Differential Equation 5, 607–623 (1993)
2. Baldi, P., Atiya, A.F.: How delays affect neural dynamics and learning. IEEE Trans. Nerual Networks 5, 612–621 (1994)

3. Van Den Driessche, P., Zou, X.F.: Global attractivity in delayed Hopfield neural networks models. SIAM J. Appl. Math. 58, 1878–1890 (1998)
4. Xu, D., Zhao, H., Zhu, H.: Global dynamics of Hopfield nerual networks invilving variable delays. Comput. Math. Appl. 42, 39–45 (2001)
5. Zhou, D., Cao, J.: Global exponential stability conditions for cellular neural networks with time-varyingdelays. Appl. Math. Comput. 131, 487–496 (2002)
6. Mohamad, S.: Global exponential stability of continuous-time and discrete-time delayed bidirectional neural networks. Phys. D 159, 233–251 (2001)
7. Chen, T.: Global exponential stability of delayed Hopfield neural networks. Neural Networks 14, 977–980 (2001)
8. Cao, J., Wang, J.: Global asymptotic stability of a general class of recurrent neural networks with time-varying delays. IEEE Trans. Circuits System I 50, 34–44 (2003)
9. Chen, Z., Ruan, J.: Global stability analysis of impulsive Cohen-Grossberg neural networks with delay. Physics Letters A 345, 101–111 (2005)
10. Yang, F., Zhang, C., Wu, D.: Global stability analysis of impulsive BAM type Cohen-Grossberg neural networks with delays. Applied Mathematics and Computation 186, 932–940 (2007)
11. Zhang, C., Yang, f., Li, W.: Global exponetial stability of BAM type Cohen-Grossberg neural network with delays and impulsive. In: Proceedings of the IEEE international Conference on Automation and Logistics, Qingdao, China, September 2008, pp. 3055–3059 (2008)
12. Bohner, M., Peterson, A.: Dynamic equations on time scales: An introduction with applications. Birkhauser, Boston (2001)
13. Lakshmikantham, V., Vatsala, A.S.: Hybrid systems ontime scales. J. Comput. Appl. Math. 141, 227–235 (2002)

Multistability of Delayed Neural Networks with Discontinuous Activations

Xiaofeng Chen[1], Yafei Zhou[2], and Qiankun Song[1]

[1] Department of Mathematics, Chongqing Jiaotong University,
Chongqing 400074, China
qiankunsong@163.com
[2] College of Mathematic and Software Science, Sichuan Normal University,
Chengdu 610066, China

Abstract. In the paper, we present the existence of m^n locally exponentially stable equilibrium points for a general n-dimensional delayed neural networks with multilevel activation functions which have m segments. Furthermore, the theory is also extended to the existence of m^n locally exponentially stable limit cycles for the n-dimensional delayed neural networks evoked by periodic external input. The results are obtained through formulating parameter conditions, which are easily verifiable and independent of the delay parameter. Two numerical examples are given to show the effectiveness of the theory.

Keywords: Neural networks, Delay, Multilevel function, Multistability, Multiperiodicity.

1 Introduction

In the past decades, neural networks (NNs) have been used in many fields such as pattern recognition, associative memory, signal processing and combinatorial optimization [1]. Mathematical analysis for dynamical behaviors of NNs is a prerequisite for most applications. Due to these, extensive attention has been paid to study the dynamical behaviors such as stability, periodic oscillation, bifurcation and chaos [2,3,4]. The theory on dynamics of NNs has been developed according to the purposes of applications.

Stability and periodic oscillation are two interesting dynamical properties of NNs. Besides stability, neuron activation states may also be periodically oscillatory around an orbit. The properties of periodic solutions are of great interest since many biological and effective activities usually requires repetition [5]. In addition, an equilibrium point can be viewed as a special case of periodic oscillation of NNs with arbitrary period. In this sense, the analysis of periodic oscillation of NNs is more general than that of equilibrium point. Moreover, multistability is a necessary property in order to enable certain applications where monostability networks could be computationally restrictive [6]. Recently, there have been extensive results on multistability for NNs [7,8,9].

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 603–610, 2010.

However, all of the studies mentioned above are focused upon NNs with activation functions being continuous or even Lipschitzen. In fact, the authors of Forti et al. have pointed out that some common NNs with discontinuous activations are important and frequently arise in a large number of practice [10]. For example, in optimal control problems, open-loop, bang-bang controllers switch discontinuously between extreme values of the inputs to generate minimum time trajectories from the initial to the final states. The dynamical behaviors of NNs with discontinuous activations have been studied in [11,12,13,14].

To the best of our knowledge, the study on multistability for NNs with discontinuous activations is few. In [15] the multistability has been developed for NNs with multilevel activations which are discontinuous but without delays. It is found that a k-neuron networks can have up to n^k locally exponentially stable equilibria. In [15] the authors point out that the dynamics of NNs with delay was their another subject of interest to investigate in near future. Motivated by the fact, in this paper, we consider the following delayed NNs with discontinuous activations,

$$\dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{n} \alpha_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} \beta_{ij} f_j(x_j(t-\tau_{ij})) + I_i, \tag{1}$$

for $i = 1, 2, \cdots, n$, where $x_i(t)$ is the state of neuron i; α_{ij} and β_{ij} are connection weights from neuron j to neuron i; $f_j(\cdot)$ is the input-output activation function of jth neuron; τ_{ij} corresponds to the transmission delay and satisfies $0 \leq \tau_{ij} \leq \tau$, where τ is a constant; I_i is a constant external input. The model of delayed NNs with periodic inputs associate with system (1) is described as follow,

$$\dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{n} \alpha_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} \beta_{ij} f_j(x_j(t-\tau_{ij})) + I_i(t), \tag{2}$$

where $I_i(t)$ is a continuous w-periodic function.

Throughout this paper, we assume that f_j are multilevel functions with m segments, which are described as follow,

$$f_j(x) = \begin{cases} b_{1j}, & \text{if } x < a_1, \\ b_{lj} & \text{if } a_{l-1} \leq x < a_l, \quad l = 2, 3, \cdots, m-1, \\ b_{mj}, & \text{if } x \geq a_{m-1}, \end{cases} \tag{3}$$

where $a_1 < a_2 < \cdots < a_{m-1}$, and $-1 = b_{1j} < b_{2j} < \cdots < b_{m-1,j} < b_{mj} = 1$. If we set $a_0 = -\infty$ and $a_m = +\infty$, (3) can be rewritten as

$$f_j(x) = b_{lj}, \quad \text{if } a_{l-1} \leq x < a_l, \quad l = 1, 2, \cdots, m$$

Here, if $a_{l-1} = -\infty$ or $a_l = +\infty$, the inequality constraints are one side. Denote

$$\Omega = \{(a_0, a_1), [a_1, a_2), [a_2, a_3), \cdots, [a_{m-1}, a_m)\}$$

and

$$\Omega^n = \left\{ \prod_{i=1}^{n} J_i \middle| J_i \in \Omega \right\}.$$

It is obvious that $\bigcup \Omega = \mathbb{R}$ and $\bigcup \Omega^n = \mathbb{R}^n$. Given a set S, denote $\mathrm{int}(S)$ is the interior of S. Given any $x \in \mathbb{R}^n$, denote $\|x\| = \max_{1\le i\le n}\{|x_i|\}$. Let $C([t_0-\tau, t_0], D)$ be the space of continuous functions mapping $[t_0-\tau, t_0]$ into $D \subset \mathbb{R}^n$ with norm defined by $\|\phi\|_{t_0} = \max_{1\le i\le n}\{\sup_{s\in[t_0-\tau,t_0]} |\phi(s)|\}$.

For any $\phi = (\phi_1, \phi_2, \cdots, \phi_n)^T \in C([t_0-\tau, t_0], \mathbb{R}^n)$, the initial conditions for the network (1) or (2) are assumed to be

$$x_i(t) = \phi_i(t), \quad t_0 - \tau \le t \le t_0, \quad i = 1, 2, \cdots, n.$$

Denote $x(t; t_0, \phi)$ as the solution of (1) (or (2)) with initial condition ϕ, which means that $x(t; t_0, \phi)$ is continuous and satisfies (1) (or (2)) and $x(t; t_0, \phi) = \phi(t)$ for $t \in [t_0-\tau, t_0]$.

2 Preliminaries

In this section, we state a definition and two preliminary lemmas, which will be used throughout this paper.

Definition 1. *A set D is said to be an invariant set of system (1), if the solution $x(t; t_0, \phi)$ of (1) with any initial condition $\phi(s) \in C([t_0-\tau, t_0], D)$, satisfies $x(t; t_0, \phi) \in D$ for $t \ge t_0$.*

Lemma 1. *If for any $i \in \{1, 2, \cdots, n\}$ and $l \in \{1, 2, \cdots, m\}$,*

$$a_{l-1} + \sum_{j=1, j\ne i}^{n} |\alpha_{ij} + \beta_{ij}| < (\alpha_{ii} + \beta_{ii}) b_{li} + I_i < a_l - \sum_{j=1, j\ne i}^{n} |\alpha_{ij} + \beta_{ij}|, \qquad (4)$$

then for any $D \in \Omega^n$, $\mathrm{int}(D)$ is an invariant set of (1).

Proof. Let $\mathrm{int}(D) = (a_{k_1-1}, a_{k_1}) \times (a_{k_2-1}, a_{k_2}) \times \cdots \times (a_{k_n-1}, a_{k_n})$, where $k_i \in \{1, 2, \cdots, m\}$. For any $\phi(s) \in C([t_0-\tau, t_0], \mathrm{int}(D))$, there must be a close set $D' \subset \mathrm{int}(D)$ such that

$$\phi(s) \in D', \quad \forall s \in [t_0-\tau, t_0].$$

Let $D' = [a'_{k_1-1}, a'_{k_1}] \times [a'_{k_2-1}, a'_{k_2}] \times \cdots \times [a'_{k_n-1}, a'_{k_n}]$. Then $a_{k_i-1} < a'_{k_i-1} < a'_{k_i} < a_{k_i}$, for any $i \in \{1, 2, \cdots, n\}$. Let $x(t; t_0, \phi)$ is a solution of (1) with the initial condition $\phi(s)$. We say that it stays in the local area D'. if it is not true, there is a time $t_2 > t_0$ such that $x(t_2)$ is not in D'. There must be an $x_p(t_2)$ not in $[a'_{k_p-1}, a'_{k_p}]$, where $p \in \{1, 2, \cdots, n\}$. If $x_p(t_2) < a'_{k_p-1}$, then there is a time $t_1 \in (t_0, t_2)$ such that $x_p(t_1) = a'_{k_p-1}$ and $\dot{x}_p(t_1) < 0$. Without loss of generality,

we assume that for any $j \in \{1, 2, \cdots, n\}$, $x_j(t) \in [a'_{k_j-1}, a'_{k_j}], \forall t \in [t_0 - \tau, t_1]$. However, on the other hand, from (1) and (4), we have

$$\begin{aligned}\dot{x}_p(t_1) &= -x_p(t_1) + \sum_{j=1}^{n} \alpha_{pj} f_j(x_j(t_1)) + \sum_{j=1}^{n} \beta_{pj} f_j(x_j(t_1 - \tau_{pj})) + I_p \\ &= -a'_{k_p-1} + (\alpha_{pp} + \beta_{pp}) b_{k_p,p} + \sum_{j=1, j\neq p}^{n} \alpha_{pj} b_{k_j,j} + \sum_{j=1, j\neq p}^{n} \beta_{pj} b_{k_j,j} + I_p \\ &> -a'_{k_p-1} + (\alpha_{pp} + \beta_{pp}) b_{k_p,p} - \sum_{j=1, j\neq p}^{n} |\alpha_{ij} + \beta_{ij}| + I_p > 0,\end{aligned}$$

which is a contradiction. Thus we have $x_p(t) \geq a'_{k_p-1}, \forall t > t_0$. In the same way, we can prove $x_p(t) \leq a'_{k_p}, \forall t > t_0$. This shows the solution $x(t; t_0, \phi)$ stays in the region D', which is also in $\mathrm{int}(D)$. So $\mathrm{int}(D)$ is an invariant set of (1).

Lemma 2. *If for any $i \in \{1, 2, \cdots, n\}$ and $l \in \{1, 2, \cdots, m\}$,*

$$a_{l-1} + \sum_{j=1, j\neq i}^{n} |\alpha_{ij} + \beta_{ij}| < (\alpha_{ii} + \beta_{ii}) b_{li} + I_i(t) < a_l - \sum_{j=1, j\neq i}^{n} |\alpha_{ij} + \beta_{ij}|, \quad (5)$$

then for any $D \in \Omega^n$, $\mathrm{int}(D)$ is an invariant set of (2).

Proof. The proof is similar to Lemma 1. So we omit it.

3 Main Results

In this section we will establish some sufficient conditions to ensure that the n-neuron NNs can have m^n locally exponentially stable equilibrium points.

Theorem 1. *If the condition (4) in Lemma 1 holds, then for any $D \in \Omega^n$, the neural network (1) has a unique equilibrium point in region $\mathrm{int}(D)$ which is locally exponentially stable. Therefore, the system (1) has m^n locally exponentially stable equilibrium points in $\mathbb{R}^n$.*

Proof. Let $\mathrm{int}(D) = (a_{k_1-1}, a_{k_1}) \times (a_{k_2-1}, a_{k_2}) \times \cdots \times (a_{k_n-1}, a_{k_n})$, where $k_i \in \{1, 2, \cdots, m\}$. If $x^* = (x_1^*, x_2^*, \cdots, x_n^*) \in \mathrm{int}(D)$ is an equilibrium of system (1), we have

$$x_i^* = \sum_{j=1}^{n} \alpha_{ij} b_{k_j,j} + \sum_{j=1}^{n} \beta_{ij} b_{k_j,j} + I_i.$$

From (4), we have

$$a_{k_i-1} < \sum_{j=1}^{n} \alpha_{ij} b_{k_j,j} + \sum_{j=1}^{n} \beta_{ij} b_{k_j,j} + I_i < a_{k_i}.$$

Hence, the system (1) has exactly equilibrium x^* in the local region $\mathrm{int}(D)$.

From Lemma 1, we know that $\text{int}(D)$ is a invariant set of (1), which means for any initial condition $\phi(s) \in C([t_0-\tau, t_0], \text{int}(D))$, the solution $x(t;t_0,\phi)$ stays in $\text{int}(D)$. Thus we can rewrite (1) as

$$\dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{n} \alpha_{ij} b_{k_j,j} + \sum_{j=1}^{n} \beta_{ij} b_{k_j,j} + I_i = -x_i(t) + x_i^*.$$

It is obvious that $x(t;t_0,\phi)$ tends to x^* exponentially.

It is noted that Ω^n has m^n elements, hence there are m^n locally exponentially stable equilibrium points for system (1).

In the following we will consider the multiperiodicity of the system (2).

Theorem 2. *If the condition (5) in Lemma 2 holds, then for any $D \in \Omega^n$, The neural network (2) has a unique limit cycle in region $\text{int}(D)$ which is locally exponentially stable. Therefore, the system (5) has m^n locally exponentially stable limit cycles in $\mathbb{R}^n$.*

Proof. Let $\text{int}(D) = (a_{k_1-1}, a_{k_1}) \times (a_{k_2-1}, a_{k_2}) \times \cdots \times (a_{k_n-1}, a_{k_n})$, where $k_i \in \{1, 2, \cdots, m\}$. By Lemma 2, $\text{int}(D)$ is a invariant set of (2), which means for any initial condition $\phi(s) \in C([t_0-\tau, t_0], \text{int}(D))$, the solution $x(t;t_0,\phi)$ stays in $\text{int}(D)$. Thus the system (2) can be rewritten as

$$\dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{n} \alpha_{ij} b_{k_j,j} + \sum_{j=1}^{n} \beta_{ij} b_{k_j,j} + I_i(t). \tag{6}$$

Denote $x_t(s;\phi) = x(t+s;t_0,\phi)$, where $s \in [t_0-\tau, t_0], t \geq 0$. It is clear that $x_t(\cdot;\phi) \in C([t_0-\tau, t_0], \text{int}(D))$ since $\text{int}(D)$ is an invariant set. Define a mapping $H : C([t_0-\tau, t_0], \text{int}(D)) \to C([t_0-\tau, t_0], \text{int}(D))$ by $H(\phi) = x_w(\cdot;\phi)$. Then $H^m(\phi) = x_{mw}(\cdot;\phi)$

Let us fix any $\xi(s), \zeta(s) \in C([t_0-\tau, t_0], \text{int}(D))$ and consider the solutions $x(t;t_0,\xi)$ and $x(t;t_0,\zeta)$ of system (2) with initial conditions $\xi(s)$ and $\zeta(s)$, respectively. Then we can derive from (6) that

$$\frac{\mathrm{d}(x_i(t;t_0,\xi) - x_i(t;t_0,\zeta))}{\mathrm{d}t} = -(x_i(t;t_0,\xi) - x_i(t;t_0,\zeta)).$$

Thus

$$x_i(t;t_0,\xi) - x_i(t;t_0,\zeta) = (\xi_i(t_0) - \zeta_i(t_0))e^{-(t-t_0)}, \tag{7}$$

which can imply that

$$\begin{aligned} \|H^m(\xi) - H^m(\zeta)\|_{t_0} &= \|(\xi(t_0) - \zeta(t_0))e^{-(mw+s-t_0)}\|_{t_0} \\ &\leq e^{-(mw-\tau)}\|\xi(t_0) - \zeta(t_0)\| \\ &\leq e^{-(mw-\tau)}\|\xi - \zeta\|_{t_0}. \end{aligned}$$

In particular, setting $m \in \mathbb{N}$ such that $mw - \tau > 0$, we have that H^m is a contraction mapping and has a unique fixed point in $C([t_0-\tau, t_0], \text{int}(D))$, that

is, there is a unique $\phi^* \in C([t_0 - \tau, t_0], \mathrm{int}(D))$ such that $H^m(\phi^*) = \phi^*$. In addition,

$$H^m(H(\phi^*)) = H(H^m(\phi^*)) = H(\phi^*)$$

that shows $H(\phi^*)$ is also a fixed point of H^m. Hence, by the uniqueness of the fixed point of H^m, we have $H(\phi^*) = \phi^*$, that is $x_w(s; \phi^*) = \phi^*(s)$. Let $x(t; t_0, \phi^*)$ be a solution of (2) with initial condition ϕ^*. From (6), by using $I_i(t+w) = I_i(t)$, it follows that $x(t + w; t_0, \phi^*)$ is also a solution of (2). Thus, $\forall t \geq t_0$,

$$x(t + w; t_0, \phi^*) = x(t; t_0, x_w(s; \phi^*)) = x(t; t_0, \phi^*).$$

This shows that $x(t; t_0, \phi^*)$ is a periodic solution with period w. From (7), we can have

$$\begin{aligned} \|x(t; t_0, \phi) - x(t; t_0, \phi^*)\| &= \|\phi(t_0) - \phi^*(t_0)\| e^{-(t-t_0)} \\ &\leq \|\phi - \phi^*\|_{t_0} e^{-(t-t_0)}. \end{aligned}$$

This means that the periodic solution $x(t; t_0, \phi^*)$ is locally exponentially stable in $\mathrm{int}(D)$.

It is noted that Ω^n has m^n elements, hence there are m^n locally exponentially stable limit cycles for system (1).

Remark 1. In [15], the authors investigated the dynamics for NNs without delays. When $\beta_{ij} = 0$ in (1) and (2), the NNs reduce to a special case without delays. Therefore, Theorem 1 and 2 extend the work of [15].

4 Illustrative Examples

In this section, we give two examples to illustrate the effectiveness of our results.

Example 1. Consider the following neural networks,

$$\begin{aligned} \begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{pmatrix} = &- \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 1.5 & -0.2 \\ 0.5 & 0.8 \end{pmatrix} \begin{pmatrix} f_1(x_1(t)) \\ f_2(x_2(t)) \end{pmatrix} \\ &+ \begin{pmatrix} -0.2 & 0.1 \\ -0.3 & 0.6 \end{pmatrix} \begin{pmatrix} f_1(x_1(t-0.1)) \\ f_2(x_2(t-0.2)) \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.15 \end{pmatrix}, \end{aligned}$$

where

$$f_1(u) = \begin{cases} -1, & \text{if } u < -1, \\ -0.5 & \text{if } -1 \leq u < 0, \\ 0.5 & \text{if } 0 \leq u < 1, \\ 1, & \text{if } u \geq 1, \end{cases} \quad f_2(u) = \begin{cases} -1, & \text{if } u < -1, \\ -0.6 & \text{if } -1 \leq u < 0, \\ 0.4 & \text{if } 0 \leq u < 1, \\ 1, & \text{if } u \geq 1. \end{cases}$$

This networks satisfies the conditions of Theorem 1. As numerical simulate shows, the networks has 4^2 locally stable points (Fig. 1).

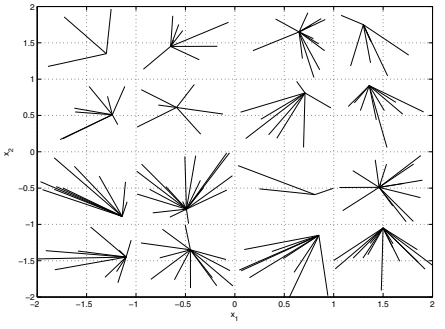


Fig. 1. Phase plot in Example 1

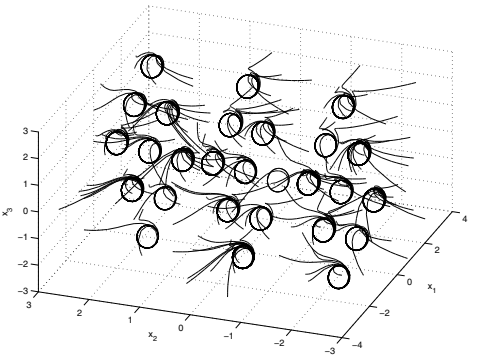


Fig. 2. Phase plot in Example 2

Example 2. Consider the following neural networks,

$$\begin{pmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\end{pmatrix} = -\begin{pmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{pmatrix} + \begin{pmatrix}1.5 & 0.2 & -0.1\\ 0.3 & 1 & 0.1\\ 0.1 & -0.2 & -0.1\end{pmatrix}\begin{pmatrix}f_1(x_1(t))\\ f_2(x_2(t))\\ f_3(x_3(t))\end{pmatrix}$$
$$+\begin{pmatrix}0.5 & 0.2 & -0.1\\ -0.1 & 1 & 0.15\\ 0.1 & 0.15 & 1.9\end{pmatrix}\begin{pmatrix}f_1(x_1(t-0.1))\\ f_2(x_2(t-0.1))\\ f_3(x_3(t-0.1))\end{pmatrix} + \begin{pmatrix}0.2\sin(t)\\ 0.3\cos(t)\\ 0.5\sin(t)\end{pmatrix},$$

where

$$f_1(u) = f_2(u) = f_3(u) = \begin{cases}-1, & \text{if } u < -1,\\ 0 & \text{if } -1 \le u < 1,\\ 1, & \text{if } u \ge 1.\end{cases}$$

This parameters also satisfy the conditions of Theorem 2. As numerical simulate shows, there are 3^3 locally stable limits cycles (Fig. 2).

5 Conclusion

In this paper, we have investigated the multistability for a class of delayed NNs with multilevel activation function. By using local invariant sets, some parameter conditions, which can be examined easily, are established to guarantee the the networks to have m^n exponentially stable equilibria and m^n exponentially stable limit cycles. Simulation examples are employed to illustrate the theories. Furthermore, considering other classes of discontinuous activation functions, which are more general than multilevel functions, is an interesting job. It will be performed in near future.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 60974132 and 10772152, Youth Foundation of Chongqing Jiaotong University.

References

1. Cao, J., Li, X.: Stability in Delayed Cohen-Grossberg Neural Networks: LMI Optimization Approach. Physica D 212, 54–65 (2005)
2. Song, Q., Cao, J.: Stability Analysis of Cohen-Crossberg Neural Networks with Both Time-varying and Continuously Distributed Delays. J. Comput. Appl. Math. 197, 188–203 (2006)
3. Yu, W., Cao, J.: Stability and Hopf Bifurcation Analysis on A Four-neuron BAM Neural Network with Time Delays. Phys. Lett. A 351, 64–78 (2006)
4. Chen, T., Lu, W., Chen, G.: Dynamical Dehaviors of A Large Class of General Delayed Neural Networks. Neural Computation 17, 949–968 (2005)
5. Townley, S., Ilchmann, A., Weiss, M.G., Mcclements, W., Ruiz, A.C., Owens, D.H., Pratzel-Wolters, D.: Existence and Learning of Oscillations in Recurrent Neural Networks. IEEE Trans. Neural Netw. 11, 205–214 (2000)
6. Zhang, L., Zhang, Y., Yu, J.: Multiperiodicity and Attractivity of Delayed Recurrent Neural Networks with Unsaturating Piecewise Linear Transfer Functions. IEEE Trans. Neural Netw. 19, 158–167 (2008)
7. Huang, G., Cao, J.: Multistability in Bidirectional Associative Memory Neural Networks. Phys. Lett. A 372, 2842–2854 (2008)
8. Nie, X., Cao, J.: Multistability of Competitive Neural Networks with Time-varying and Distributed Delays. Nonlinear Analysis: Real World Applications 10, 928–942 (2009)
9. Shih, C.W., Tseng, J.P.: Convergent Dynamics for Multistable Delayed Neural Networks. Nonlinearity 21, 2361–2389 (2008)
10. Forti, M., Nistri, P.: Global Convergence of Neural Networks with Discontinuous Neuron Activations. IEEE Trans. Circuits Syst. I 50, 1421–1435 (2003)
11. Forti, M., Grazzini, M., Nistri, P., Pancioni, L.: Generalized Lyapunov Approach for Convergence of Neural Networks with Discontinuous or Non-Lipschitz Activations. Physica D 214, 88–99 (2006)
12. Huang, L., Guo, Z.: Global Convergence of Periodic Solution of Neural Networks with Discontinuous Activation Functions. Chaos, Solitons and Fractals 42, 2351–2356 (2009)
13. Li, L., Huang, L.: Dynamical Behaviors of A Class of Recurrent Neural Networks with Discontinuous Neuron Activations. Appl. Math. Modell. 33, 4326–4336 (2009)
14. Lu, W., Chen, T.: Dynamical Behaviors of Cohen-Grossberg Neural Networks with Discontinuous Activation Functions. Neural Netw. 18, 231–242 (2005)
15. Huang, G., Cao, J.: Multistability of Neural Networks with Discontinuous Activation Function. Comm. Nonlinear Sci. Numer. Simulat. 13, 2279–2289 (2008)

Finite-Time Boundedness Analysis of Uncertain CGNNs with Multiple Delays

Xiaohong Wang[1], Minghui Jiang[1], Chuntao Jiang[2], and Shengrong Li[1]

[1] Institute of Nonlinear and Complex System, China Three Gorges University, YiChang, Hubei 443002, China
[2] Department of Mathematics, Northwestern University, Xi'an 710127, China

Abstract. The problem of finite-time boundedness(FTB) for a class of Cohen-Grossberg neural networks(CGNNs) with multiple delays and parameter perturbations is analyzed in this paper. By way of extending the concept of FTB for time delay system and constructing a suitable Lyapunov function and using linear matrix inequality(LMI) technique, some delay-dependent criteria are derived to guarantee FTB for uncertain and certain CGNNs with multiple delays. Meanwhile, an algorithm is also presented. Finally, simulation examples are given to demonstrate the effectiveness of the conclusion.

Keywords: Cohen-Grossberg neural network; Parameter perturbations; Finite-time boundedness; Linear matrix inequality; Multiple delays.

1 Introduction

Recently, linear matrix inequality(LMI)-based techniques have been successfully used to tackle various stability problems for neural networks with a single delay[1] and multiple delays[2]. The main advantages of the developed approach include that firstly it only needs tuning of parameters and matrices, and secondly it can be solved numerically using the effective interior-point algorithm that can solve optimization problems involving LMI[3]. Since Cohen and Grossberg proposed a class of neural networks in 1983[4], and this model has received increasing interest[5-6] due to its promising for applications in classification, parallel computing, associative memory. In many practical applications, some systems may be unstable, in this case, the main concern is the behavior of the system over a fixed finite time interval, it could be required that the trajectories of the controlled system do not exceed given bounds. In order to deal with this problem, Peter Dorato[7] presented the concept of finite-time stability(FTS).

In this paper, we further extend the results of FTB to the general Cohen-Grossberg neural networks with multiple delays and norm-boundedness parametric uncertainties in Section 2. The sufficient conditions are derived for ensuring FTB of the uncertain and certain CGNNs in Section 3. Section 4 gives some numerical examples to demonstrate the main theoretical results. We conclude this paper in Section 5.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 611–618, 2010.

2 Problem Statement

In this brief, considering the influences of uncertainties for a class of generalized Cohen-Grossberg neural networks with multiple delays, we can describe it by the following nonlinear differential equation:

$$\dot{u}(t) = -\alpha(u(t))[\beta(u(t)) - (W_0 + \Delta W_0)g(u(t)) - \sum_{i=1}^{K}(W_k + \Delta W_k)g(u(t-\tau_k))], \tag{1}$$

where $u = (u_1, \cdots, u_n)^T \in R^n$, $\alpha(u) = \mathrm{diag}(\alpha_1(u_1), \cdots, \alpha_n(u_n))$, $\beta(u) = (\beta_1(u_1), \cdots, \beta_n(u_n))^T$. Amplification function $\alpha_i(\cdot)$, behaved function $\beta_i(\cdot)$ and activation function $g_i(\cdot)$ are subject to certain conditions to be specified later. $W_0 = [w_{0ij}] \in R^{n\times n}$ means that part of the interconnecting structure which is not associated with delay, $W_k = [w_{kij}] \in R^{n\times n}$ denotes that part of the interconnecting structure which is associated with delay τ_k, where τ_k stand for the kth delay, $k = 1, \cdots, K$ and $\tau = \max\{\tau_1, \cdots, \tau_k\}$. $\Delta W_k = [\Delta w_{kij}] \in R^{n\times n}$, $k = 0, \cdots, K$ are time-invariant matrices representing the norm-bounded uncertainties.

The initial condition is $u(s) = \phi(s)$, for $s \in [-\tau, 0]$, where $\phi \in C([-\tau, 0], \Omega)$. Here, $C([-\tau, 0], \Omega)$ denotes the Banach space of continuous vector-valued functions mapping the interval $[-\tau, 0]$ into R^n with a topology of uniform convergence.

In order to establish the conditions of main results for the neural networks (1), it is necessary to make the following assumptions:

Assumption 1. For the uncertainties ΔW_k, $k = 0, \cdots, K$, we assume that

$$[\Delta W_0, \cdots, \Delta W_K] = HF[E_0, \cdots, E_K], \tag{2}$$

where F is an unknown matrix representing parametric uncertainty, which satisfies

$$F^T F \leq I, \tag{3}$$

and $H, E_0, \cdots, E_K$ are known constant matrices with appropriate dimensions.

Assumption 2. For $i = 1, 2, \cdots, n$, we assume that

(H1) $\alpha_i(u_i) > 0$, α_i are bounded, that is, there exist positive constants α_i^m and α_i^M such that $0 < \alpha_i^m \leq \alpha_i(u_i) \leq \alpha_i^M < +\infty$ for all $u \in R$;

(H2) The behaved function $\beta_i(\cdot) \in C(R, R)$, and $(\beta_i(x) - \beta_i(y))/(x - y) \geq \gamma_i > 0$ for any $x, y \in R$ and $x \neq y$;

(H3) The activation function $g(u)$ is bounded and global Lipschitz with Lipschitz constants $\sigma_i \in R^+$, that is $|g_i(x) - g_i(y)| \leq \sigma_i|x - y|$ for any $x, y \in R$;

Lemma 1. Assume that (H1)-(H3) hold, then system (1) has at least an equilibrium point.

The proof is omitted.

Definition 1. System (1) is said to be finite time boundedness(FTB) with respect to $(c_1, c_2, T)(0 < c_1 < c_2)$, if

$$\sup_{t\in[-\tau,0]} \|\varphi^T(t)\|^2 \le c_1^2 \Rightarrow \|x^T(t)\|^2 < c_2^2, \quad \forall t \in [0, T].$$

Lemma 2. [8] For matrices Y, D and E of appropriate dimensions, where Y is a symmetric matrix, then $Y + DFE + E^T F^T D^T < 0$ holds for all matrix F satisfying $F^T F \le I$, if and only if there exists a constant $\varepsilon > 0$ such that $Y + \varepsilon DD^T + \varepsilon^{-1} E^T E < 0$ holds.

3 Main Results

In this section, sufficient conditions for FTB are given by Theorem 1 and Corollary 1.

Theorem 1. For any bounded delay τ_k, $k = 1, \cdots, K$, system (1) is FTB with respect to (c_1, c_2, T), if there exist a nonnegative scalar α, a positive definite matrix P, a positive constant ε and positive definite diagonal matrices Q, Λ_k, $k = 1, \cdots, K$, such that the following conditions:

$$\begin{bmatrix} \Gamma & PA^M W_0 & PA^M W_1 & \cdots & PA^M W_K & PA^M H \\ W_0^T A^M P & -Q + \varepsilon E_0^T E_0 & \varepsilon E_0^T E_1 & \cdots & \varepsilon E_0^T E_K & 0 \\ W_1^T A^M P & \varepsilon E_1^T E_0 & -\Lambda_1 + \varepsilon E_1^T E_1 & \cdots & \varepsilon E_1^T E_K & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ W_K^T A^M P & \varepsilon E_K^T E_0 & \varepsilon E_K^T E_1 & \cdots & -\Lambda_K + \varepsilon E_K^T E_K & 0 \\ H^T A^M P & 0 & 0 & \cdots & 0 & -\varepsilon I \end{bmatrix} < 0 \quad (4)$$

and

$$\frac{e^{\alpha T} c_1^2 [\lambda_{\max}(P) + \tau \Lambda \lambda_{\max}(\sigma^T \sigma)]}{\lambda_{\min}(P)} < c_2^2 \quad (5)$$

hold, where $\Gamma = -B^T A^m P - PA^m B + \sigma(\sum_{i=1}^K \Lambda_k + Q)\sigma - \alpha P$, $\sigma = \mathrm{diag}(\sigma_1, \cdots, \sigma_{\mathrm{n}})$, $A^M = \mathrm{diag}(\alpha_1^{\mathrm{M}}, \cdots, \alpha_{\mathrm{n}}^{\mathrm{M}})$, $A^m = \mathrm{diag}(\alpha_1^{\mathrm{m}}, \cdots, \alpha_{\mathrm{n}}^{\mathrm{m}})$, $B = \mathrm{diag}(\gamma_1, \cdots, \gamma_{\mathrm{n}})$, $\Lambda = \sum_{k=1}^K (\lambda_{\max}(\Lambda_k))$.

Proof. Here, we introduce the following Lyapunov function:

$$V(x(t)) = x^T(t) P x(t) + \sum_{k=1}^K \int_{t-\tau_k}^t f^T(x(s)) \Lambda_k f(x(s)) \mathrm{ds}.$$

Then, the derivative of $V(x(t))$ with respect to t along any trajectory of system (1) is given by

$$\dot{V}(x(t)) \le -x^T(t)(B^T A^m P + PA^m B)x(t) + 2x^T(t) PA^M (W_0 + \Delta W_0) f(x(t))$$

$$+2x^T(t)PA^M\sum_{k=1}^{K}(W_k+\Delta W_k)f(x(t-\tau_k))+\sum_{k=1}^{K}f^T(x(t))\Lambda_k f(x(t))$$
$$-\sum_{k=1}^{K}f^T(x(t-\tau_k))\Lambda_k f(x(t-\tau_k))+f^T(x(t))Qf(x(t))$$
$$-f^T(x(t))Qf(x(t)), \tag{6}$$

Noting that $Q>0$ is a diagonal matrix and Assumption 2, we can obtain

$$f^T(x(t))Qf(x(t))\le x^T(t)\sigma^TQ\sigma x(t). \tag{7}$$

Taking (7) into (6), we can get

$$\dot V(x(t))\le\xi^T\Theta\xi,$$

where $\xi=[x^T(t),f^T(x(t)),f^T(x(t-\tau_1)),\cdots,f^T(x(t-\tau_K))]^T$,

$$\Theta=\begin{bmatrix}\Gamma_1 & PA^M(W_0+\Delta W_0) & PA^M(W_1+\Delta W_1) & \cdots & PA^M(W_K+\Delta W_K)\\ * & -Q & 0 & \cdots & 0\\ * & * & -\Lambda_1 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ * & * & * & \cdots & -\Lambda_K\end{bmatrix},$$

and $\Gamma_1=-B^TA^mP-PA^mB+\sigma(\sum_{i=1}^{K}\Lambda_k+Q)\sigma$.

Making use of Assumption 1, from the form of Θ we can rewrite it as follows:

$$\Theta=\begin{bmatrix}\Gamma_1 & PA^MW_0 & PA^MW_1 & \cdots & PA^MW_K\\ W_0^TA^MP & -Q & 0 & \cdots & 0\\ W_1^TA^MP & 0 & -\Lambda_1 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ W_K^TA^MP & 0 & 0 & \cdots & -\Lambda_K\end{bmatrix}$$
$$+\begin{bmatrix}PA^MH\\0\\0\\\vdots\\0\end{bmatrix}F\begin{bmatrix}0\ E_0\ E_1\cdots E_K\end{bmatrix}$$
$$+\begin{bmatrix}0\\E_0^T\\E_1^T\\\vdots\\E_K^T\end{bmatrix}F^T\begin{bmatrix}H^TA^MP\,0\,0\cdots0\end{bmatrix}. \tag{8}$$

From Lemma 2, $\Theta < 0$ is equivalent to the following inequality:

$$\Upsilon = \begin{bmatrix} \Gamma_1 & PA^M W_0 & PA^M W_1 & \cdots & PA^M W_K \\ W_0^T A^M P & -Q & 0 & \cdots & 0 \\ W_1^T A^M P & 0 & -\Lambda_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ W_K^T A^M P & 0 & 0 & \cdots & -\Lambda_K \end{bmatrix}$$
$$+\varepsilon^{-1} \begin{bmatrix} PA^M H \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} H^T A^M P \, 0 \, 0 \cdots 0 \end{bmatrix} + \varepsilon \begin{bmatrix} 0 \\ E_0^T \\ E_1^T \\ \vdots \\ E_K^T \end{bmatrix} \begin{bmatrix} 0 \, E_0 \, E_1 \cdots E_K \end{bmatrix}$$
$$= \begin{bmatrix} \Gamma_2 & PA^M W_0 & PA^M W_1 & \cdots & PA^M W_K \\ W_0^T A^M P & -Q + \varepsilon E_0^T E_0 & \varepsilon E_0^T E_1 & \cdots & \varepsilon E_0^T E_K \\ W_1^T A^M P & \varepsilon E_1 E_0^T & -\Lambda_1 + \varepsilon E_1^T E_1 & \cdots & \varepsilon E_1^T E_K \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ W_K^T A^M P & \varepsilon E_K^T E_0 & \varepsilon E_K^T E_1 & \cdots & -\Lambda_K + \varepsilon E_K^T E_K \end{bmatrix} < 0,$$

where $\Gamma_2 = -B^T A^m P - PA^m B + \sigma(\sum_{i=1}^K \Lambda_k + Q)\sigma + \varepsilon^{-1} PA^M H H^T A^M P$.

Continuing to using Schur complement, $\Upsilon < 0$ is equivalent to the following LMI:

$$\Upsilon_1 = \begin{bmatrix} \Gamma_1 & PA^M W_0 & PA^M W_1 & \cdots & PA^M W_K & PA^M H \\ W_0^T A^M P & -Q + \varepsilon E_0^T E_0 & \varepsilon E_0^T E_1 & \cdots & \varepsilon E_0^T E_K & 0 \\ W_1^T A^M P & \varepsilon E_1^T E_0 & -\Lambda_1 + \varepsilon E_1^T E_1 & \cdots & \varepsilon E_1^T E_K & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ W_K^T A^M P & \varepsilon E_K^T E_0 & \varepsilon E_K^T E_1 & \cdots & -\Lambda_K + \varepsilon E_K^T E_K & 0 \\ H^T A^M P & 0 & 0 & \cdots & 0 & -\varepsilon I \end{bmatrix} < 0.$$

Then, we have

$$\Upsilon = \Upsilon_1 + \Upsilon_2,$$

where $\Upsilon_2 = \mathrm{diag}[\alpha \mathrm{P}, 0, 0, \cdots, 0]$.

Furthermore, we can also deduce that

$$\begin{aligned} &\dot{V}(x(t)) \le \xi^T \Theta \xi < 0 \Leftrightarrow \dot{V}(x(t)) \le \xi^T \Upsilon \xi < 0 \\ \Leftrightarrow\ &\dot{V}(x(t)) \le \xi^T (\Upsilon_1 + \Upsilon_2) \xi < 0 \Rightarrow \dot{V}(x(t)) \le \xi^T \Upsilon_2 \xi < 0. \end{aligned}$$

Namely,

$$\begin{aligned} \dot{V}(x(t)) &< \alpha x^T(t) P x(t) \le \alpha [x^T(t) P x(t) + \sum_{k=1}^K \int_{t-\tau_k}^t f^T(x(s)) \Lambda_k f(x(s)) \mathrm{ds}] \\ &= \alpha V(x(t)). \end{aligned} \tag{9}$$

Multiplying (9) by $e^{-\alpha t}$, we can get

$$\frac{\mathrm{d}}{\mathrm{dt}}(e^{-\alpha t}V) < 0, \tag{10}$$

Integrating (10) from 0 to t, with $t \in [0, T]$, we have

$$e^{-\alpha t}V(x(t)) \leq V(x(0)).$$

Then

$$\begin{aligned} V(x(t)) &\leq e^{\alpha t}V(x(0)) = e^{\alpha t}[x^T(0)Px(0) + \sum_{k=1}^{K}\int_{-\tau_k}^{0} f^T(x(s))\Lambda_k f(x(s))\mathrm{ds}] \\ &\leq e^{\alpha T}c_1^2[\lambda_{\max}(P) + \tau\Lambda\lambda_{\max}(\sigma^T\sigma)]. \end{aligned} \tag{11}$$

Noting that

$$x^T(t)Px(t) \leq V(x(t)) \Rightarrow \lambda_{\min}(P)x^T(t)x(t) \leq V(x(t)). \tag{12}$$

Putting together (11) and (12), we have

$$\|x^T(t)\|^2 \leq \frac{e^{\alpha T}c_1^2[\lambda_{\max}(P) + \tau\Lambda\lambda_{\max}(\sigma^T\sigma)]}{\lambda_{\min}(P)}.$$

Condition (9) and the above inequality imply

$$\|x^T(t)\|^2 < c_2^2, \quad \text{for all}\, t \in [0, T].$$

The proof is completed.

When $\Delta W_k = 0$, $k = 0, \cdots, K$, we have the following corollary.

Corollary 1. For any bounded delay τ_k, $k = 1, \cdots, K$, system (1) is FTB with respect to (c_1, c_2, T), if there exist a nonnegative scalar α, a positive definite matrix P and positive definite diagonal matrices Q, Λ_k, $k = 1, \cdots, K$, such that the following conditions:

$$\begin{bmatrix} \Gamma & PA^M W_0 & PA^M W_1 & \cdots & PA^M W_K \\ W_0^T A^M P & -Q & 0 & \cdots & 0 \\ W_1^T A^M P & 0 & -\Lambda_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ W_K^T A^M P & 0 & 0 & \cdots & -\Lambda_K \end{bmatrix} < 0 \tag{13}$$

and

$$\frac{e^{\alpha T}c_1^2[\lambda_{\max}(P) + \tau\Lambda\lambda_{\max}(\sigma^T\sigma)]}{\lambda_{\min}(P)} < c_2^2 \tag{14}$$

hold, where $\Gamma = -B^T A^m P - PA^m B + \sigma(\sum_{i=1}^{K}\Lambda_k + Q)\sigma - \alpha P$, $\sigma = \mathrm{diag}(\sigma_1, \cdots, \sigma_{\mathrm{n}})$, $A^M = \mathrm{diag}(\alpha_1^{\mathrm{M}}, \cdots, \alpha_{\mathrm{n}}^{\mathrm{M}})$, $A^m = \mathrm{diag}(\alpha_1^{\mathrm{m}}, \cdots, \alpha_{\mathrm{n}}^{\mathrm{m}})$, $B = \mathrm{diag}(\gamma_1, \cdots, \gamma_{\mathrm{n}})$, $\Lambda = \sum_{k=1}^{K}(\lambda_{\max}(\Lambda_k))$.

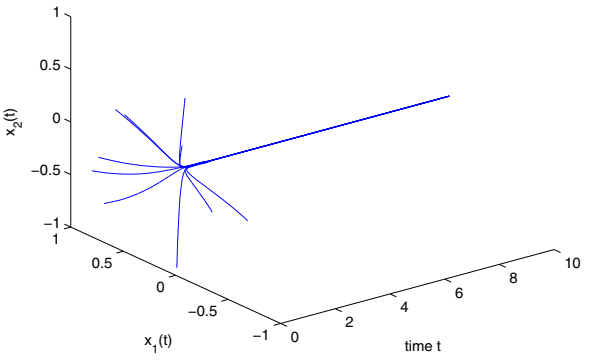


Fig. 1. The behavior of system (15) with ten state trajectories of random initials

4 Simulation Examples

In this section, we give four numerical examples to illustrate the validity of our theoretical results by using MATLAB's LMI toolbox.

Firstly, considering the case that there exist two delays τ_1, τ_2 and certain parameters, then system (1) can be rewritten as

$$\begin{aligned}\dot{x}(t) = -a(x(t))[b(x(t)) - W_0 f(x(t)) - W_1 f(x(t-\tau_1)) \\ -W_2 f(x(t-\tau_2))],\end{aligned} \tag{15}$$

Example 1. For (15), we choose the parameters as

$$W_0 = \begin{bmatrix} -2.5 & 0 \\ 1 & 0.5 \end{bmatrix}, W_1 = \begin{bmatrix} 1 & 0.8 \\ 0 & -1 \end{bmatrix}, W_2 = \begin{bmatrix} -2.3 & 1.7 \\ 1 & 0 \end{bmatrix},$$
$$a(x(t)) = \begin{bmatrix} \sin(x_1) + 1.2 & 0 \\ 0 & \cos(x_2) + 1.3 \end{bmatrix},$$

$f(x(t)) = [0.13\tanh(x_1), 0.13\tanh(x_2)]^T$, $b(x(t)) = [10\tan(x_1), 6\tan(x_2)]^T$.

So, by computing, we can get

$$0.2 \le a_1(x_1) \le 2.2, 0.3 \le a_1(x_1) \le 2.3.$$

Hence,

$A^M = \mathrm{diag}[2.2, 2.3]$, $A^m = \mathrm{diag}[0.2, 0.3]$, $B = \mathrm{diag}[10, 6]$, $\sigma = \mathrm{diag}[0.13, 0.13]$, $\tau_1 = 0.8$, $\tau_2 = 1.0$, $c_1 = 1$.

If $\alpha = 0$, by Corollary 1, we can get the neural network system (15) is global asymptotically stable with the minimum boundedness $c_2 = 4$. The solution is given as follows:

$$P = \begin{bmatrix} 0.3978 & 0.5352 \\ 0.5352 & 1.4503 \end{bmatrix}, Q = \begin{bmatrix} 8.5969 & 0 \\ 0 & 4.7541 \end{bmatrix}, \Lambda_1 = \begin{bmatrix} 3.0899 & 0 \\ 0 & 8.0907 \end{bmatrix},$$
$$\Lambda_2 = \begin{bmatrix} 8.0449 & 0 \\ 0 & 9.5577 \end{bmatrix}.$$

During the process of simulation, we choose the different initial conditions. The following simulation picture of system (15) testifies the validity of result. That is to say, system (15) is global asymptotically stable when $\tau_1 = 0.8s$, $\tau_2 = 1.0s$.

In this case, system (15) is also FTB with respect to (c_1, c_2, T) for a maximum $T = 3.7s$, when $\alpha = 0.1$, $c_2 = 4$.

5 Conclusion

The paper has mainly studied the problem of FTB analysis for uncertain Cohen-Grossberg neural network with multiple delays. Based on LMI technique, some sufficient conditions which guarantee the FTB for uncertain and certain CGNNs, respectively, are derived. In the end, an example has been provided to demonstrate the validity of the proposed theoretical results.

Acknowledgments

The work was supported by the Scientific Innovation TeamProject of Hubei Provincial Department of Education (T200809), the Doctoral Start-up Funds of China Three Gorges University(0620060061) and Hubei Natural Science Foundation (No.2008CDZ046).

References

1. Liao, X., Chen, G., Sanchez, E.N.: Delay-dependent exponential stability analysis of delayed neural networks: an LMI approach. Neural Networks 15, 855–866 (2002)
2. Ji, C., Zhang, H., Wei, Y.: LMI approach for global robust stability of Cohen-Grossberg neural networks with multiple delays. Neurocom. 71, 475–485 (2008)
3. Boyd, S., EI Ghaoui, L., Feron, E., Balakrishnan, V.: Linear Matrix Inequalities in System and Control Theory. SIAM, Philadelphia (1994)
4. Cohen, M., Grossberg, S.: Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. IEEE Trans. Syst. Man and Cybernetics 13, 815–826 (1983)
5. Song, Q., Cao, J.: Global exponential robust stability of Cohen-Grossberg neural network with time-varying delays and reaction-diffusion terms. J. Franklin Institute 343, 705–719 (2006)
6. Cao, J., Song, Q.: Stability in Cohen-Grossberg type BAM neural networks with time-varying delays. Nonlinearity 19(7), 1601–1617 (2006)
7. Dorato, P.: Short time stability in linear time-varying system. In: Proc. IRE International Convention Record, Part 4, pp. 83–87 (1961)
8. Boyd, S., El GhaouiL, E., Feron, E., Balakrishnan, V.: Linear matrix inequality in system and control Theory. SIAM Studies in Applied Mathematics. SIAM, Philadelphia (1994)

Dissipativity Analysis of Stochastic Neural Networks with Time-Varying Delays

Jianting Zhou[1], Qiankun Song[2], and Jianxi Yang[1]

[1] College of Civil Engineering and Architecture, Chongqing Jiaotong University, Chongqing 400074, China
[2] Department of Mathematics, Chongqing Jiaotong University, Chongqing 400074, China
qiankunsong@163.com

Abstract. In this paper, the problem on global dissipativity is investigated for stochastic neural networks with time-varying delays and generalized activation functions. By constructing appropriate Lyapunov-Krasovskii functionals and employing stochastic analysis method and linear matrix inequality (LMI) technique, a new delay-dependent criterion for checking the global dissipativity of the addressed neural networks is established in terms of LMIs, which can be checked numerically using the effective LMI toolbox in MATLAB. The proposed dissipativity criterion does not require the monotonicity of the activation functions and the differentiability of the time-varying delays, which means that our result generalizes and further improves those in the earlier publications. An example is given to show the effectiveness and less conservatism of the obtained conditions.

Keywords: Stochastic neural networks, Global dissipativity, Time-varying delays.

1 Introduction

The dissipativity in dynamical systems is a more general concept and it has found applications in the areas such as stability theory, chaos and synchronization theory, system norm estimation, and robust control [1]. Recently, the dissipativity of neural networks was considered [2], some sufficient conditions on the dissipativity of delayed neural networks were derived, for example, see [2]-[9] and references therein. In [2]-[4], authors analyzed the dissipativity of neural network with constant delays, and derived several sufficient conditions for the global dissipativity of neural network with constant delays. In [5]-[6], authors considered the global dissipativity and global robust dissipativity for neural network with both time-varying delays and unbounded distributed delays, several sufficient conditions for checking the global dissipativity and global robust dissipativity were obtained. In [7]-[8], by using linear matrix inequality technique, authors investigated the global dissipativity of neural network with both discrete time-varying delays and distributed time-varying delays. In [9], authors proposed the

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 619–626, 2010.

concept on global dissipativity of stochastic neural networks with constant delay, and gave several criteria for checking the global dissipativity of stochastic neural networks with constant delay. However, time delays can occur in an irregular fashion, and sometimes the time-varying delays are not differentiable. In such a case, the method developed in [9] may be difficult to be applied, and it is therefore necessary to further investigate the global dissipativity problem of stochastic neural networks with time-varying delays under *milder* assumptions for the activation functions. To the best of our knowledge, few authors have considered the problem on the dissipativity of stochastic neural networks with time-varying delays and generalized activation functions. Therefore, the study on the dissipativity of stochastic neural networks with with time-varying delays and generalized activation functions is not only important but also necessary.

Motivated by the above discussions, the objective of this paper is to study the problem on global dissipativity for stochastic neural networks with time-varying delays and generalized activation functions. By employing appropriate Lyapunov-Krasovskii functionals and using stochastic analysis method and LMI technique, we obtain a new sufficient condition for checking the global dissipativity of the addressed neural networks.

2 Problem Formulation and Preliminaries

In this paper, we consider the following neural network:

$$\begin{aligned} dx(t) = [-Cx(t) + Af(x(t)) + Bf(x(t-\tau(t))) + u]dt \\ +\sigma(t, x(t), x(t-\tau(t)))d\omega(t) \end{aligned} \tag{1}$$

for $t \geq 0$, where $x(t) = (x_1(t), x_2(t), \cdots, x_n(t))^T \in R^n$ is state vector of the network at time t, n corresponds to the number of neurons; $C = \mathrm{diag}(c_1, c_2, \cdots, c_n)$ is a positive diagonal matrix, $A = (a_{ij})_{n\times n}$ and $B = (b_{ij})_{n\times n}$ are interconnection weight matrices; $f(x(t)) = (f_1(x_1(t)), f_2(x_2(t)), \cdots, f_n(x_n(t)))^T$ denotes neuron activation at time t; $u = (u_1, u_2, \cdots, u_n)^T$ is a external input vector; $\tau(t)$ is the time-varying delay, and is assumed to satisfy $0 \leq \tau(t) \leq \tau$, where τ is constant. $\sigma(t, x(t), x(t-\tau(t))) \in \mathbb{R}^{n\times n}$ is the diffusion coefficient matrix and $\omega(t) = (\omega_1(t), \omega_2(t), \cdots, \omega_n(t))^T$ is an n-dimensional Brownian motion defined on a complete probability space $(\Omega, F, \{F_t\}_{t\geq 0}, \mathcal{P})$ with a filtration $\{F_t\}_{t\geq 0}$ satisfying the usual conditions (i.e., it is right continuous and F_0 contains all P-null sets). The initial condition associated with model (1) is given by $x(s) = \phi(s)$ on $s \in [-\tau, 0]$, and $\phi \in L^2_{F_0}([-\tau, 0], \mathbb{R}^n)$.

Throughout this paper, we make the following assumptions:

(H1).([10]) For any $j \in \{1, 2, \cdots, n\}$, $f_j(0) = 0$ and there exist constants F_j^- and F_j^+ such that

$$F_j^- \leq \frac{f_j(\alpha_1) - f_j(\alpha_2)}{\alpha_1 - \alpha_2} \leq F_j^+$$

for all $\alpha_1 \neq \alpha_2$.

(**H2**).([9]) There exist two symmetric positive definite matrices $R_1 > 0$ and $R_2 > 0$ such that the following inequality

$$\text{trace}[\sigma^T(t,u,v)\sigma(t,u,v)] \leq u^T R_1 u + v^T R_2 v$$

holds for all $(t,u,v) \in \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n$.

Definition 1. *([9]) Neural networks (1) is said to be a globally dissipative, if there exists a compact set $S \subseteq \mathbb{R}^n$, such that $\forall \phi_0 \in L^2_{F_0}([-\tau,0],\mathbb{R}^n)$, $\exists$ $T(\phi_0) > 0$, when $t \geq t_0 + T(\phi_0)$, $\mathbb{E}x(t,t_0,\phi_0) \subseteq S$, where $x(t,t_0,\phi_0)$ denotes the trajectory of (2) from initial state ϕ_0 and initial time t_0. In this case, S is called a globally attractive set. A set S is called a positive invariant, if $\forall \mathbb{E}\phi_0 \in S$ implies $\mathbb{E}x(t,t_0,\phi_0) \subseteq S$ for $t \geq t_0$.*

To prove our result, the following lemmas that can be found in [8] and [9] are necessary.

Lemma 1. *([8]) The inequality $2a^T b \leq a^T Q a + b^T Q^{-1} b$ holds for any a, $b \in \mathbb{R}^n$ and any $Q > 0$.*

Lemma 2. *([8]) Given constant matrices P, Q and R, where $P^T = P$, $Q^T = Q$, then*

$$\begin{bmatrix} P & R \\ R^T & -Q \end{bmatrix} < 0$$

is equivalent to the following conditions

$$Q > 0 \quad \text{and} \quad P + RQ^{-1}R^T < 0.$$

Lemma 3. *(Jensen's inequality [9]) Lete $\Omega, F, \mathcal{P}$) be a probability space, X an integrable real-valued random variable and $\Psi : \mathbb{R} \longrightarrow \mathbb{R}$ a measurable convex function, with $\mathbb{E}(|\Psi(X)|) < \infty$. Then $\Psi(\mathbb{E}(X)) < \mathbb{E}(\Psi(X))$.*

3 Main Results

For presentation convenience, in the following, we denote

$$F_1 = \text{diag}(F_1^- F_1^+, \cdots, F_n^- F_n^+), \quad F_2 = \text{diag}(\frac{F_1^- + F_1^+}{2}, \cdots, \frac{F_n^- + F_n^+}{2}),$$

Theorem 1. *Suppose that (**H**1) and (**H**2) hold. If there exist a scalar $\lambda > 0$, six symmetric positive definite matrices P_i $(i = 1,2,\cdots,6)$, two positive diagonal matrices L and S, and four matrices Q_i $(i = 1,2,3,4)$ such that the following LMIs hold:*

$$P_1 < \lambda I, \tag{2}$$

$$\Omega = \begin{bmatrix} \Omega_{11} & \Omega_{12} & Q_1 & 0 & \Omega_{15} & Q_4B & Q_1 & Q_1 & Q_2 & Q_4 & 0 & 0 \\ * & \Omega_{22} & 0 & 0 & Q_3A & Q_3B & 0 & 0 & 0 & 0 & Q_3 & 0 \\ * & * & \Omega_{33} & Q_2 & 0 & F_2S & 0 & 0 & 0 & 0 & 0 & Q_2 \\ * & * & * & -P_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & -L & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & * & -S & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\tau}P_3 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & * & * & * & -P_1 & 0 & 0 & 0 & 0 \\ * & * & * & * & * & * & * & * & -P_1 & 0 & 0 & 0 \\ * & * & * & * & * & * & * & * & * & -P_5 & 0 & 0 \\ * & * & * & * & * & * & * & * & * & * & -P_4 & 0 \\ * & * & * & * & * & * & * & * & * & * & * & -\frac{1}{\tau}P_3 \end{bmatrix} < 0, \quad (3)$$

where $\Omega_{11} = -Q_1 - Q_1^T + P_2 - F_1L - Q_4C - CQ_4^T + P_6 + \lambda(1+\tau)R_1$, $\Omega_{12} = P_1 - Q_4 - CQ_3^T$, $\Omega_{15} = F_2L + Q_4A$, $\Omega_{22} = \tau P_3 - Q_3 - Q_3^T$, $\Omega_{33} = -Q_2 - Q_2^T - F_1S + \lambda(1+\tau)R_2$, *then model (1) is globally dissipative, and*

$$S = \left\{ z : \quad |z| \le \sqrt{\frac{u^T(P_4+P_5)u}{\lambda_{min}(P_6)}}, \quad z \in \mathbb{R}^n \right\} \quad (4)$$

is a positive invariant and globally attractive set.

Proof. Let $y(t) = -Cx(t) + Af(x(t)) + Bf(x(t-\tau(t))) + u$, $\alpha(t) = \sigma(t, x(t), x(t-\tau(t)))$, then model (1) is rewritten as

$$dx(t) = y(t)dt + \alpha(t)d\omega(t). \quad (5)$$

Consider the following Lyapunov-Krasovskii functional as

$$V(t, x(t)) = x^T(t)P_1x(t) + \int_{t-\tau}^{t} x^T(s)P_2x(s)ds + \int_{-\tau}^{0}\int_{t+\theta}^{t} y^T(s)P_3y(s)ds + \int_{-\tau}^{0}\int_{t+\theta}^{t} \text{trace}[\alpha^T(s)P_1\alpha(s)]dsd\theta. \quad (6)$$

By Itô differential rule, the stochastic derivative of $V(t)$ along the trajectory of model (1) can be obtained as

$$\begin{aligned} dV(t, x(t)) = \Big\{ & 2x^T(t)P_1y(t) + x^T(t)P_2x(t) - x^T(t-\tau)P_2x(t-\tau) \\ & + \tau y^T(t)P_3y(t) - \int_{t-\tau}^{t} y^T(s)P_3y(s)ds \\ & + (1+\tau)\text{trace}[\alpha^T(t)P_1\alpha(t)] - \int_{t-\tau}^{t} \text{trace}[\alpha^T(s)P_1\alpha(s)]ds \Big\} dt \\ & + [x^T(t)P_1\alpha(t) + \alpha^T(t)P_1x(t)]d\omega(t). \end{aligned} \quad (7)$$

Integrating both sides of (5) from $t-\tau(t)$ to t, by Lemma 1 and noting $\tau(t) \le \tau$, we have

$$\begin{aligned}
0 &= -2x^T(t)Q_1\Big[x(t) - x(t-\tau(t)) - \int_{t-\tau(t)}^{t} y(s)ds - \int_{t-\tau(t)}^{t} \alpha(s)d\omega(s)\Big] \\
&\le x^T(t)\Big(- 2Q_1 + \tau Q_1 P_3^{-1} Q_1^T + Q_1 P_1^{-1} Q_1^T\Big)x(t) \\
&\quad +2x^T(t)Q_1 x(t-\tau(t)) + \int_{t-\tau(t)}^{t} y^T(s)P_3 y(s)ds \\
&\quad +\Big(\int_{t-\tau(t)}^{t} \alpha(s)d\omega(s)\Big)^T P_1 \Big(\int_{t-\tau(t)}^{t} \alpha(s)d\omega(s)\Big). \qquad (8)
\end{aligned}$$

Integrating both sides of (5) from $t-\tau$ to $t-\tau(t)$, by using of the same way, we get

$$\begin{aligned}
0 &\le x^T(t-\tau(t))\Big(- 2Q_2 + \tau Q_2 P_3^{-1} Q_2^T\Big)x(t-\tau(t)) + 2x^T(t-\tau(t))Q_2 x(t-\tau) \\
&\quad + \int_{t-\tau}^{t-\tau(t)} y^T(s)P_3 y(s)ds + x^T(t)Q_2 P_1^{-1} Q_2^T x(t) \\
&\quad +\Big(\int_{t-\tau}^{t-\tau(t)} \alpha(s)d\omega(s)\Big)^T P_1 \Big(\int_{t-\tau}^{t-\tau(t)} \alpha(s)d\omega(s)\Big). \qquad (9)
\end{aligned}$$

From assumption (**H1**), we have [10]

$$\begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix}^T \begin{bmatrix} F_1 L & -F_2 L \\ -F_2 L & L \end{bmatrix} \begin{bmatrix} x(t) \\ f(x(t)) \end{bmatrix} \le 0 \qquad (10)$$

and

$$\begin{bmatrix} x(t-\tau(t)) \\ f(x(t-\tau(t))) \end{bmatrix}^T \begin{bmatrix} F_1 S & -F_2 S \\ -F_2 S & S \end{bmatrix} \begin{bmatrix} x(t-\tau(t)) \\ f(x(t-\tau(t))) \end{bmatrix} \le 0. \qquad (11)$$

From the definition of $y(t)$ and Lemma 1, we have

$$\begin{aligned}
0 &= 2\Big(y^T(t)Q_3 + x^T(t)Q_4\Big)\Big[- y(t) - Cx(t) + Af(x(t)) \\
&\quad +Bf(x(t-\tau(t))) + u\Big] \\
&\le x^T(t)(-2Q_4 C + Q_4 P_5^{-1} Q_4^T)x(t) - 2x^T(t)(Q_4 + CQ_3^T)y(t) \\
&\quad +2x^T(t)Q_4 Af(x(t)) + 2x^T(t)Q_4 Bf(x(t-\tau(t))) \\
&\quad +y^T(t)(-2Q_3 + Q_3 P_4^{-1} Q_3^T)y(t) + 2y^T(t)Q_3 Af(x(t)) \\
&\quad +2y^T(t)Q_3 Bf(x(t-\tau(t))) + u^T(P_4 + P_5)u. \qquad (12)
\end{aligned}$$

From assumption (**H2**) and condition (2), we get

$$\text{trace}[\alpha^T(t)P_1\alpha(t)] \le \lambda[x^T(t)R_1 x(t) + x^T(t-\tau(t))R_2 x(t-\tau(t))]. \qquad (13)$$

From the proof of [11], we have

$$\begin{aligned}
&\mathbb{E}\Big\{\Big(\int_{t-\tau}^{t-\tau(t)}\alpha(s)d\omega(s)\Big)^T P_1\Big(\int_{t-\tau}^{t-\tau(t)}\alpha(s)d\omega(s)\Big)\Big\}\\
&=\mathbb{E}\Big\{\int_{t-\tau}^{t-\tau(t)}\mathrm{trace}[\alpha^T(s)P_1\alpha(s)]ds\Big\}, \qquad (14)\\
&\mathbb{E}\Big\{\Big(\int_{t-\tau(t)}^{t}\alpha(s)d\omega(s)\Big)^T P_1\Big(\int_{t-\tau(t)}^{t}\alpha(s)d\omega(s)\Big)\Big\}\\
&=\mathbb{E}\Big\{\int_{t-\tau(t)}^{t}\mathrm{trace}[\alpha^T(s)P_1\alpha(s)]ds\Big\}. \qquad (15)
\end{aligned}$$

Taking the mathematical expectation on both sides of (7), it follows from (7) to (15) that

$$d\mathbb{E}V(t,x(t)) \le \mathbb{E}\Big\{-x^T(t)P_6x(t)+u^T(P_4+P_5)u+\xi^T(t)\Pi\xi(t)\Big\}dt, \quad (16)$$

where $\xi(t)=\Big(x^T(t),y^T(t),x^T(t-\tau(t)),x^T(t-\tau),f^T(x(t)),f^T(x(t-\tau(t)))\Big)^T$, and

$$\Pi=\begin{bmatrix}
\Pi_1 & P_1-Q_4-CQ_3^T & Q_1 & 0 & F_2L+Q_4A & Q_4B\\
* & \Pi_2 & 0 & 0 & Q_3A & Q_3B\\
* & * & \Pi_3 & Q_2 & 0 & F_2S\\
* & * & * & -P_2 & 0 & 0\\
* & * & * & * & -L & 0\\
* & * & * & * & * & -S
\end{bmatrix}$$

with $\Pi_1=-Q_1-Q_1^T+\tau Q_1P_3^{-1}Q_1^T+Q_1P_1^{-1}Q_1^T+P_2+Q_2P_1^{-1}Q_2^T-F_1L-Q_4C-CQ_4^T+Q_4P_5^{-1}Q_4^T+P_6+\lambda(1+\tau)R_1$, $\Pi_2=\tau P_3-Q_3-Q_3^T+Q_3P_4^{-1}Q_3^T$, $\Pi_3=-Q_2-Q_2^T+\tau Q_2P_3^{-1}Q_2^T-F_1S+\lambda(1+\tau)R_2$.

It is easy to verify the equivalence of $\Pi<0$ and $\Omega<0$ by using Lemma 2. Thus, one can derive from condition (3), inequality (17) and Lemma 3 that

$$\begin{aligned}
d\mathbb{E}V(t,x(t)) &\le \mathbb{E}\Big\{-x^T(t)P_6x(t)+u^T(P_4+P_5)u\Big\}dt\\
&\le \Big\{-\lambda_{min}(P_6)|\mathbb{E}x(t)|^2+u^T(P_4+P_5)u\Big\}dt<0 \qquad (17)
\end{aligned}$$

when $\mathbb{E}x(t)\in\mathbb{R}^n\backslash S$. Hence,

$$\mathbb{E}V(t,x(t)) < \mathbb{E}V(t_0,x(t_0))$$

From (6), we can get that

$$\mathbb{E}V(t,x(t)) \ge \lambda_{min}(P_1)\mathbb{E}|x(t)|^2 \ge \lambda_{min}(P_1)|\mathbb{E}x(t)|^2.$$

So we can easily see that there exists a $T(t_0,\varphi_0)$ such that $\mathbb{E}x(t)\in S$ for $t>t_0+T(t_0,\varphi_0)$. By Definition 1, the stochastic neural network (1) is globally dissipative and S is an attractive set in mean of it. The proof is completed.

Remark 1. When the time-varying delay $\tau(t)$ of model (1) is not differentiable, the method developed in [9] may be difficult to derive the dissipativity criterion of model (1). To overcome the difficult, we structure a new Lyapunov-Krasovskii functional (6) in this paper. So, the obtained result is new.

4 An Example

Consider a two-neuron neural network (1), where

$$C = \begin{bmatrix} 0.4 & 0 \\ 0 & 0.3 \end{bmatrix}, \quad A = \begin{bmatrix} -0.1 & 0.3 \\ -0.1 & 0.2 \end{bmatrix}, \quad B = \begin{bmatrix} -0.4 & 0.2 \\ 0.3 & 0.1 \end{bmatrix}, \quad u = \begin{bmatrix} 1.2 \\ -0.9 \end{bmatrix},$$

$$f_1(z) = \tanh(0.2z), \quad f_2(z) = \tanh(-0.1z), \quad \tau(t) = 0.2|\sin t|.$$

$$\sigma(t, x(t), x(t-\tau(t))) = \begin{bmatrix} 0.1x_1(t) + 0.3x_1(t-\tau(t)) & 0 \\ 0 & -0.2x_2(t) + 0.1x_2(t-\tau(t)) \end{bmatrix}.$$

It can be verified that assumptions (**H1**) and (**H2**) are satisfied, and $F_1 = 0$, $F_2 = \text{diag}\{0.1, -0.05\}$, $R_1 = \begin{bmatrix} 0.04 & 0 \\ 0 & 0.06 \end{bmatrix}, R_1 = \begin{bmatrix} 0.12 & 0 \\ 0 & 0.03 \end{bmatrix}, \tau = 0.2$.

By the Matlab LMI Control Toolbox, we find a solution to the LMIs (2) and (3) as follows:

$$P_1 = \begin{bmatrix} 97.6256 & 0.6636 \\ 0.6636 & 88.4531 \end{bmatrix}, P_2 = \begin{bmatrix} 30.7742 & 0.4516 \\ 0.4516 & 18.0988 \end{bmatrix}, P_3 = \begin{bmatrix} 42.5011 & -0.1326 \\ -0.1326 & 38.8319 \end{bmatrix},$$

$$P_4 = \begin{bmatrix} 139.7012 & 0.3138 \\ 0.3138 & 140.5565 \end{bmatrix}, P_5 = \begin{bmatrix} 381.9800 & -4.2125 \\ -4.2125 & 367.5476 \end{bmatrix}, P_6 = \begin{bmatrix} 5.6510 & 0.0938 \\ 0.0938 & 4.9914 \end{bmatrix},$$

$$Q_1 = \begin{bmatrix} 9.3411 & 0.9634 \\ -1.7783 & 10.1721 \end{bmatrix}, Q_2 = \begin{bmatrix} 22.0716 & 0.2684 \\ 0.2797 & 13.7402 \end{bmatrix}, Q_3 = \begin{bmatrix} 40.0101 & 6.2630 \\ -6.8178 & 46.9272 \end{bmatrix},$$

$$Q_4 = \begin{bmatrix} 90.0020 & -2.6613 \\ 2.8820 & 79.7409 \end{bmatrix}, L = \begin{bmatrix} 124.9855 & 0 \\ 0 & 214.9204 \end{bmatrix}, S = \begin{bmatrix} 230.4348 & 0 \\ 0 & 167.3793 \end{bmatrix},$$

$$\lambda = 117.8144.$$

Therefore, by Theorem 1, we know that the considered model (1) is globally dissipative. It is easy to compute that the positive invariant and global attractive set are

$$S = \Big\{ z : \quad |z| \le 15.3383, \quad z \in \mathbb{R}^n \Big\}.$$

It should be pointed out that the conditions in [9] can not be applied to check the dissipativity for this example since it require that the delay is constant, and activation functions are monotonic or bounded.

5 Conclusions

In this paper, the global dissipativity has been investigated for stochastic neural networks with time-varying delays and generalized activation functions. By constructing appropriate Lyapunov-Krasovskii functionals and employing stochastic analysis method and linear matrix inequality (LMI) technique, a new delay-dependent criterion for checking the global dissipativity of the addressed neural networks has been established in terms of LMIs, which can be checked numerically using the effective LMI toolbox in MATLAB. The obtained result generalizes and improves the earlier publications. An example has been provided to demonstrate the effectiveness and less conservatism of the proposed criterion.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 60974132 and 50608072, and in part by Natural Science Foundation Project of CQ CSTC2008BA6038 and 2008BB2351, and Scientific & Technological Research Projects of CQKJ090406, and the Ministry of Education for New Century Excellent Talent Support Program.

References

1. Hale, J.: Asymptotic Behavior of Dissipative Systems. American Mathematical Society, New York (1989)
2. Liao, X.X., Wang, J.: Global Dissipativity of Continuous-time Recurrent Neural Networks with Time Delay. Physics Review E 68, 1–7 (2003)
3. Arik, S.: On The Global Dissipativity of Dynamical Neural Networks with Time Delays. Physics Letters A 326, 126–132 (2004)
4. Song, Q.K., Zhao, Z.J.: Global Dissipativity of Neural Networks with Both Variable and Unbounded Delays. Chaos, Solitons and Fractals 25, 393–401 (2005)
5. Song, Q.K., Zhao, Z.J.: Global Robust Dissipativity of Interval Neural Networks with Both Variable and Unbounded Delays. Dynamics of Continuous, Discrete and Impulsive Systems 14, 355–369 (2007)
6. Lou, X.Y., Cui, B.T.: Global Robust Dissipativity for Integro-differential Systems Modeling Neural Networks with Delays. Chaos, Solitons and Fractals 36, 469–478 (2008)
7. Cao, J.D., Yuan, K., Ho, D.W.C., Lam, J.: Global Point Dissipativity of Neural Networks with Mixed Time-varying Delay. Chaos 16, 013105 (2006)
8. Song, Q.K., Cao, J.D.: Global Dissipativity Analysis on Uncertain Neural Networks with Mixed Time-varying Delays. Chaos 18, 043126 (2008)
9. Wang, G.J., Cao, J.D., Wang, L.: Global Dissipativity of Stochastic Neural Networks with Time Delay. Journal of the Franklin Institute 346, 794–807 (2009)
10. Liu, Y.R., Wang, Z.D., Liu, X.H.: Global Exponential Stability of Generalized Recurrent Neural Networks with Discrete and Distributed Delays. Neural Networks 19, 667–675 (2006)
11. Su, W.W., Chen, Y.M.: Global Asymptotic Stability Analysis for Neutral Stochastic Neural Networks with Time-Varying Delays. Communications in Nonlinear Science and Numerical Simulation 14, 1576–1581 (2009)

Multistability Analysis: High-Order Networks Do Not Imply Greater Storage Capacity Than First-Order Ones

Zhenkun Huang

School of Sciences, Jimei University, Xiamen 361021, P.R. China

Abstract. This paper presents new multistability of the networks with high-order interconnection. We construct invariant regions and establish new criteria of coexistence of equilibria which are exponentially stable. Our results show that high-order interactions of neurons may have lower storage capacity than first-order ones. Numerical simulations will illustrate our new and interesting results.

Keywords: Multistability, high-order networks, division regions, exponential stability.

1 Introduction

The analog neural networks are widely investigated due to their important applications such pattern recognition, associative memories and optimization [1] and the dynamical behavior has been discussed extensively [2,3,4,5,6,7,8,9,10] and so on. It should be noted that authors in [2] pointed out that neural models with high-order synaptic connectivity would improve dramatically their storage capacity. Meanwhile, in many previous works such as [3,4], authors claim that one of their main motivations of research of high-order neural networks relies on the basic facts: High-order neural networks have greater storage capacity and higher fault tolerance than lower-order ones. Hence, an interesting and serious question arises: Does high-order synaptic connectivity always imply faster convergence and greater storage capacity?

Strongly motivated by the question, we should consider the following general class of networks with high-order synaptic connectivity:

$$\frac{du_k(t)}{dt} = -b_k u_k(t) + \sum_{j=1}^{N} w_{kj} g_j\big(u_j(t)\big) + \sum_{j=1}^{N}\sum_{l=1}^{N} e_{kjl} g_j\Big[u_j(t-\tau_{kj}(t))\Big] g_l\Big[u_l(t-\tau_{kl}(t))\Big] + I_k, \tag{1}$$

where $k, j, l \in \mathcal{N} := \big\{1, 2, \cdots, N\big\}$. We will derive some criteria guaranteeing the existence and exponential stability of equilibria for any designated region and show that our results drop some conservative assumptions which should be satisfied in [5-8], [10] and improve the existing ones.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 627–634, 2010.

2 Preliminaries

Denote by $\mathcal{C}\big([t_0-\tau,t_0],\mathbb{R}^N\big)$ the space of all continuous functions mapping $[t_0-\tau,t_0]$ into $\mathbb{R}^N$ with the norm $\|\Phi\|=\max_{k\in\mathcal{N}}\big[\sup_{t_0-\tau\le s\le t_0}|\phi_k(s)|\big]$, where $\Phi=\big(\phi_1,\phi_2,\cdots,\phi_N\big)\in\mathcal{C}\big([t_0-\tau,t_0],\mathbb{R}^N\big)$, $0<\tau<+\infty$. Let $u^{\Phi}(t)$ be the solution of system (1) with $\Phi\in\mathcal{C}\big([t_0-\tau,t_0],\mathbb{R}^N\big)$. For each k, j, $l\in\mathcal{N}$, we assume that

(H_1) $b_k>0$, w_{kj}, e_{kjl} and I_k are constants, $w_{kk}\ge 0$, $0\le\tau_{kj}(t)\le\tau:=\max_{k,j\in\mathcal{N}}\{\tau_{kj}\}$.

(H_2) Each $g_j(\cdot)$ satisfies $g_j(\cdot)\in C^2$, $\dot g_j(\cdot)>0$ and $\lim_{\upsilon\to-\infty}g_j(\upsilon)=\check g_j<g_j(\upsilon)<\hat g_j=\lim_{\upsilon\to+\infty}g_j(\upsilon)<+\infty$. Moreover, there is a $\varpi_j\ge 0$ such that $(\upsilon-\varpi_j)\ddot g_j(\upsilon)<0$ for all $\upsilon\in\mathbb{R}/\{\varpi_j\}$.

The assumption (H_2) implies that each $g_j(\cdot)$ has basic properties:

(P_1) $|g_j(\upsilon)|\le g_j^{\natural}:=\max\{|\hat g_j|,|\check g_j|\}$ for all $\upsilon\in\mathbb{R}$.

(P_2) $\sup_{\upsilon\in\mathbb{R}}\dot g_j(\upsilon)=\dot g_j(\varpi_j)$ and $\lim_{\upsilon\to\pm\infty}\dot g_j(\upsilon)=\inf_{\upsilon\in\mathbb{R}}\dot g_j(\upsilon)=0$.

(P_3) $\dot g_j(\upsilon)$ is strictly increasing on $(-\infty,\varpi_j]$ and is strictly decreasing on $[\varpi_j,+\infty)$.

Define the following functions ($k=1,2,\cdots,N$):

$$F_k^{\flat}(\upsilon):=(-1)^{\flat}\left[\sum_{j\ne k}|w_{kj}|g_j^{\natural}+\sum_{j=1}^{N}\sum_{l=1}^{N}|e_{kjl}|g_j^{\natural}g_l^{\natural}\right]+F_k(\upsilon)+I_k,\tag{2}$$

where $F_k(\upsilon):=-b_k\upsilon+w_{kk}g_k(\upsilon)$ and $\flat=1,2$. Since $b_k>0$ and $w_{kk}\ge 0$ hold, there are only two cases for us to discuss:

Case I. $(H_{g_k}^D)$ $b_k<w_{kk}\sup\limits_{\upsilon\in\mathbb{R}}\dot g_k(\upsilon)=w_{kk}\dot g_k(\varpi_k)$.

Similarly as [7-8], if $(H_{g_k}^D)$ holds then there exist only two points p_k and q_k with $p_k<\varpi_k<q_k$ such that $\dot F_k(p_k)=\dot F_k(q_k)=0$ and $\dot F_k(\upsilon)\cdot sgn\big\{(\upsilon-p_k)(\upsilon-q_k)\big\}<0$, where $\upsilon\in\mathbb{R}$ and $\upsilon\ne p_k,q_k$. Due to $\dot F_k(\upsilon)<0$ for $\upsilon\in(-\infty,p_k)$ and $\lim_{\upsilon\to-\infty}F_k^2(\upsilon)=+\infty$, if $F_k^2(p_k)<0$, then there exists a unique point $p_k^{\dagger}$ with $p_k^{\dagger}<p_k$ such that $F_k^2(p_k^{\dagger})=0$ and $F_k^2(\upsilon)<0$ for $\upsilon\in(p_k^{\dagger},p_k]$. Similarly, due to $\dot F_k(\upsilon)<0$ for $\upsilon\in(q_k,+\infty)$ and $\lim_{\upsilon\to+\infty}F_k^1(\upsilon)=-\infty$, if $F_k^1(q_k)>0$, then there exists a unique point $q_k^{\dagger}$ with $q_k<q_k^{\dagger}$ such that $F_k^1(q_k^{\dagger})=0$ and $F_k^1(\upsilon)>0$ for $\upsilon\in[q_k,q_k^{\dagger})$.

Case II. $(H_{g_k}^U)$ $b_k\ge w_{kk}\sup\limits_{\upsilon\in\mathbb{R}}\dot g_k(\upsilon)=w_{kk}\dot g_k(\varpi_k)$.

It is obvious that $\dot F_k(\upsilon)\le 0$, $\dot F_k(\upsilon)=0$ if and only if $\dot g_k(\upsilon)=b_k/w_{kk}$ and $\upsilon=\varpi_k$. Due to $\lim_{\upsilon\to\pm\infty}F_k^2(\upsilon)=\mp\infty$ and the continuity of $F_k^2(\upsilon)$, if $\dot F_k(\upsilon)<0$ ($\upsilon\ne\varpi_k$), then there exists only one point $q_k^{\ddagger}$ such that $F_k^2(q_k^{\ddagger})=0$ and $F_k^2(\upsilon)<0$ for all $\upsilon\in(q_k^{\ddagger},+\infty)$. Similar argument can derive that there exists only one point $p_k^{\ddagger}$ with $p_k^{\ddagger}<q_k^{\ddagger}$ such that $F_k^1(p_k^{\ddagger})=0$ and $F_k^1(\upsilon)>0$ for all $\upsilon\in(-\infty,p_k^{\ddagger})$.

Definition 1. $\widetilde{\mathcal{N}} := (\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3)$ *is said to be a division of* $\mathcal{N}$ *if and only if (i)*$\mathcal{N}_2 \subseteq \big\{k \in \mathcal{N}\big|(H^U_{g_k})$ *holds*$\big\}$, $\mathcal{N}_1 \subseteq \big\{k \in \mathcal{N}\big|(H^D_{g_k})$ *holds and* $F^2_k(p_k) < 0\big\}$, $\mathcal{N}_3 \subseteq \big\{k \in \mathcal{N}\big|(H^D_{g_k})$ *holds and* $F^1_k(q_k) > 0\big\}$*; (ii)*$\mathcal{N}_1 \cap \mathcal{N}_3$ *is empty and* $\mathcal{N}_1 \cup \mathcal{N}_2 \cup \mathcal{N}_3 = \mathcal{N}$.

For each division $\widetilde{\mathcal{N}} = (\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3)$ of $\mathcal{N}$, we always let $\Sigma_{\widetilde{\mathcal{N}}} := \Big\{u \in \mathbb{R}^N\big|u_k \in (-\infty, p^\dagger_k],\ k \in \mathcal{N}_1;\ u_k \in [p^\ddagger_k, q^\ddagger_k],\ k \in \mathcal{N}_2;\ u_k \in [q^\dagger_k, +\infty),\ k \in \mathcal{N}_3\Big\}$ and $\Sigma_{\widetilde{\mathcal{N}}}$ is said be a division region of $\mathbb{R}^N$ with respect to the division $\widetilde{\mathcal{N}}$. Without otherwise statement, we always designate $\mathfrak{v} = 1, 2, 3$ and $\mathfrak{o}^1_j := p^\dagger_j$ for $j \in \mathcal{N}_1$, $\mathfrak{o}^2_j := \mathfrak{o}_j$ for $j \in \mathcal{N}_2$, $\mathfrak{o}^3_j := q^\dagger_j$ for $j \in \mathcal{N}_3$.

3 Main Results

Theorem 1. *Assume that* (H_1)*-*(H_2) *hold. For each division* $\widetilde{\mathcal{N}} = (\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3)$ *of* $\mathcal{N}$*, if*

$$\begin{aligned} 2b_k - \sum_{j=1}^{N}\left[|w_{kj}| + \sum_{l=1}^{N}(|e_{jkl}| + |e_{jlk}|)g^\natural_l\right]\dot{g}_k(\mathfrak{o}^i_k) \\ > \sum_{\mathfrak{v}=1}^{3}\sum_{j\in\mathcal{N}_\mathfrak{v}}\left[|w_{kj}| + \sum_{l=1}^{N}(|e_{kjl}| + |e_{klj}|)g^\natural_l\right]\dot{g}_j(\mathfrak{o}^\mathfrak{v}_j),\ \ k \in \mathcal{N}_i \end{aligned} \tag{3}$$

hold, then it follows that (a) Each $\Sigma_{\widetilde{\mathcal{N}}}$ *is a positively invariant set of system (1); (b) System (1) has an equilibrium in* $\Sigma_{\widetilde{\mathcal{N}}}$ *which is locally exponentially stable.*

Proof. (a) Consider any solution $u^\Phi(t) = (u^\Phi_1(t), \cdots, u^\Phi_N(t))$ with initial condition $\Phi = (\phi_1, \cdots, \phi_N) \in \mathcal{C}\big([t_0 - \tau, t_0], \Sigma_{\widetilde{\mathcal{N}}}\big)$. We claim that the solution $u^\Phi(t) \in \Sigma_{\widetilde{\mathcal{N}}}$ for all $t \geq t_0$. Contrarily, there are three cases for us to discuss:

Case 1. As $k \in \mathcal{N}_1$ and there exist a sufficiently small constant $\epsilon > 0$ ($\epsilon \ll p_k - p^\dagger_k$) and $t_1 > t_0$ such that $u^\Phi_k(t_1) = p^\dagger_k + \epsilon$, $\dot{u}^\Phi_k(t_1) \geq 0$ and $u^\Phi_k(t) \leq p^\dagger_k$ for $t_0 - \tau \leq t < t_1$. Due to $w_{kk} \geq 0$, (H_2), Case I of Section 2 and local monotonicity of $g_k(\cdot)$, it follows from system (1) that

$$\dot{u}^\Phi_k(t_1) \leq F_k(p^\dagger_k + \epsilon) + \sum_{j\neq k}|w_{kj}|g^\natural_j + \sum_{j=1}^{N}\sum_{l=1}^{N}|e_{kjl}|g^\natural_j g^\natural_l + I_k = F^2_k(p^\dagger_k + \epsilon) < 0.$$

This contradicts that $\dot{u}^\Phi_k(t_1) \geq 0$. Therefore $u^\Phi_k(t) \leq p^\dagger_k$ for $t \geq t_0$, $k \in \mathcal{N}_1$.

Case 2. As $k \in \mathcal{N}_2$ and there exist a sufficiently small constant $\epsilon > 0$ ($\epsilon \ll q^\ddagger_k - p^\ddagger_k$) and $t_1 > t_0$ such that either $u^\Phi_k(t_1) = q^\ddagger_k + \epsilon$, $\dot{u}^\Phi_k(t_1) \geq 0$ and $u^\Phi_k(t) \leq q^\ddagger_k$ for $t_0 - \tau \leq t < t_1$ or $u^\Phi_k(t_1) = p^\ddagger_k - \epsilon$, $\dot{u}^\Phi_k(t_1) \leq 0$ and $u^\Phi_k(t) \geq p^\ddagger_k$ for $t_0 - \tau \leq t < t_1$. We only consider the first subcase. From system (1), due to $w_{kk} \geq 0$, (H_2), Case II of Section 2 and local monotonicity of $g_k(\cdot)$, we get

$$\dot{u}^\Phi_k(t_1) \leq F_k\big(q^\ddagger_k + \epsilon\big) + \sum_{j\neq k}|w_{kj}|g^\natural_j + \sum_{j=1}^{N}\sum_{l=1}^{N}|e_{kjl}|g^\natural_j g^\natural_l + I_k = F^2_k\big(q^\ddagger_k + \epsilon\big) < 0.$$

This contradicts that $\dot{u}_k^{\Phi}(t_1) \geq 0$. Therefore $u_k^{\Phi}(t) \leq q_k^{\ddagger}$ for $t \geq t_0$, $k \in \mathcal{N}_2$. The second subcase can be proved similarly. Hence we have $p_k^{\ddagger} \leq u_k^{\Phi}(t) \leq q_k^{\ddagger}$ for $t \geq t_0$, $k \in \mathcal{N}_2$.

Case 3. As $k \in \mathcal{N}_3$ and there exist a sufficiently small constant $\epsilon > 0$ ($\epsilon \ll q_k^{\dagger} - q_k$) and $t_1 > t_0$ such that $u_k^{\Phi}(t_1) = q_k^{\dagger} - \epsilon$, $\dot{u}_k^{\Phi}(t_1) \leq 0$ and $u_k^{\Phi}(t) \geq q_k^{\dagger}$ for $t_0 - \tau \leq t < t_1$. From system (1), due to $w_{kk} \geq 0$, (H_2), Case I of Section 2 and local monotonicity of $g_k(\cdot)$, we get

$$\dot{u}_k^{\Phi}(t_1) \geq F_k(q_k^{\dagger} - \epsilon) - \sum_{j \neq k} |w_{kj}| g_j^{\natural} - \sum_{j=1}^{N} \sum_{l=1}^{N} |e_{kjl}| g_j^{\natural} g_l^{\natural} + I_k = F_k^1(q_k^{\dagger} - \epsilon) > 0.$$

This contradicts that $\dot{u}_k^{\Phi}(t_1) \leq 0$. Therefore $u_k^{\Phi}(t) \geq q_k^{\dagger}$ for $t \geq t_0$, $k \in \mathcal{N}_3$. From Case 1 to Case 3, we know that $\Sigma_{\widetilde{\mathcal{N}}}$ is a positively invariant set of system (1).

(b)Let $u^{\Phi}(t)$ and $u^{\Psi}(t)$ be any two solutions of system (1) with $\Phi(s), \Psi(s) \in \mathcal{C}\big([t_0 - \tau, t_0], \Sigma_{\widetilde{\mathcal{N}}}\big)$, respectively. By the positive invariance of $\Sigma_{\widetilde{\mathcal{N}}}$, we get that $u^{\Phi}(t), u^{\Psi}(t) \in \Sigma_{\widetilde{\mathcal{N}}}$ for all $t \geq t_0$. Define $\mathfrak{w}_k(t) = u_k^{\Phi}(t) - u_k^{\Psi}(t)$, $k \in \mathcal{N}$ and $t \geq t_0$. It follows from system (1) that

$$\begin{aligned} \frac{d\mathfrak{w}_k(t)}{dt} &= -b_k \mathfrak{w}_k(t) + \sum_{j=1}^{N} w_{kj} \Big[g_j(u_j^{\Phi}(t)) - g_j(u_j^{\Psi}(t)) \Big] \\ &\quad + \sum_{j=1}^{N} \sum_{l=1}^{N} e_{kjl} \Big[g_j\big(u_j^{\Phi}(t - \tau_{kj}(t))\big) g_l\big(u_l^{\Phi}(t - \tau_{kl}(t))\big) \\ &\quad - g_j\big(u_j^{\Psi}(t - \tau_{kj}(t))\big) g_l\big(u_l^{\Psi}(t - \tau_{kl}(t))\big) \Big], \end{aligned} \tag{4}$$

Define $\mathscr{H}(\mathfrak{w})(t) := \sum_{k=1}^{N} \mathfrak{w}_k(t)^2/2$ for $t \geq t_0$, where $\mathfrak{w}(t) = (\mathfrak{w}_1(t), \mathfrak{w}_2(t), \cdots, \mathfrak{w}_N(t))$. From (4), applying $a^2 + b^2 \geq 2ab$, we compute that

$$\begin{aligned} \frac{d}{dt}\mathscr{H}(\mathfrak{w})(t) &\leq -\sum_{k=1}^{N} \Bigg\{ b_k - \sum_{j=1}^{N} \frac{|w_{jk}|}{2} \dot{g}_k(\mathfrak{a}_k^{i(k)}) - \sum_{\mathfrak{v}=1}^{3} \sum_{j \in \mathcal{N}_{\mathfrak{v}}} \Bigg[\sum_{l=1}^{N} \frac{(|e_{kjl}| + |e_{klj}|) g_l^{\natural}}{2} \\ &\quad + \frac{|w_{kj}|}{2} \Bigg] \dot{g}_j(\mathfrak{a}_j^{\mathfrak{v}}) \Bigg\} \mathfrak{w}_k(t)^2 + \sum_{k=1}^{N} \sum_{j=1}^{N} \frac{\sum_{l=1}^{N} (|e_{jkl}| + |e_{jlk}|) g_l^{\natural}}{2} \dot{g}_k(\mathfrak{a}_k^{i(k)}) \\ &\quad \times \sup_{s \in [t-\tau, t]} \mathfrak{w}_k(s)^2 \leq -\mathfrak{a} \mathscr{H}(\mathfrak{w})(t) + \mathfrak{B} \sup_{s \in [t-\tau, t]} \mathscr{H}(\mathfrak{w})(s), \end{aligned}$$

where $\dot{g}_k(\mathfrak{a}_k^{i(k)}) := \dot{g}_k(\mathfrak{a}_k^i)$ as $k \in \mathcal{N}_i$ and

$$2\mathfrak{a} = \min_{1 \leq k \leq N} \Bigg\{ 2b_k - \sum_{j=1}^{N} |w_{jk}| \dot{g}_k(\mathfrak{a}_k^{i(k)})$$

$$-\sum_{\upsilon=1}^{3}\sum_{j\in\mathcal{N}_\upsilon}\left[|w_{kj}|+\sum_{l=1}^{N}(|e_{kjl}|+|e_{klj}|)g_l^\natural\right]\dot{g}_j(\mathfrak{a}_j^\upsilon)\Bigg\},$$

$$2\beta=\max_{1\le k\le N}\left\{\sum_{j=1}^{N}\sum_{l=1}^{N}(|e_{jkl}|+|e_{jlk}|)g_l^\natural\dot{g}_k(\mathfrak{a}_k^{i(k)})\right\}.$$

It follows from (3) that

$$\mathscr{H}(\mathfrak{w})(t)\le\sup_{s\in[-\tau,0]}\mathscr{H}(\mathfrak{w})(s)\exp(-\lambda\, t),\qquad t\ge 0 \tag{5}$$

where λ is the unique solution of $\lambda=\mathfrak{a}-\beta\, e^{\lambda\,\tau}$. Then it follows

$$\sum_{k=1}^{N}\mathfrak{w}_k(t)^2\le\sup_{s\in[-\tau,0]}\sum_{k=1}^{N}\mathfrak{w}_k(s)^2\exp(-\lambda\, t),\qquad t\ge 0.$$

By the Cauchy convergence principle, $u^\Phi(t)$ approaches a unique constant vector u^* which is an equilibrium of system (1) in $\Sigma_{\widetilde{\mathcal{N}}}$. From (5), we know that u^* is locally exponentially stable in $\Sigma_{\widetilde{\mathcal{N}}}$.

Remark 1. In [5-8] and [10], $(H_{g_k}^D)$, $F_k^2(p_k)<0$ and $F_k^1(q_k)>0$ should be satisfied for all $k\in\mathcal{N}$. These conservative ones are relaxed in our theorem.

Theorem 2. *Assume that* (H_1)-(H_2) *hold,* $\widetilde{\mathcal{N}}=(\mathcal{N}_1,\mathcal{N}_2,\mathcal{N}_3)$ *is a division of* $\mathcal{N}$ *and*

$$\sum_{j=1}^{N}|w_{kj}|g_j^\natural+\sum_{l=1}^{N}\sum_{j=1}^{N}|e_{kjl}|g_j^\natural g_l^\natural<\begin{cases}b_kp_k-I_k, & k\in\mathcal{N}_1\\ \\ -b_kq_k+I_k, & k\in\mathcal{N}_3\end{cases} \tag{6}$$

Reset

$$p_k^\dagger:=\frac{I_k+\sum\limits_{j=1}^{N}|w_{kj}|g_j^\natural+\sum\limits_{l=1}^{N}\sum\limits_{j=1}^{N}|e_{kjl}|g_j^\natural g_l^\natural}{b_k},\quad q_k^\dagger:=\frac{I_k-\sum\limits_{j=1}^{N}|w_{kj}|g_j^\natural-\sum\limits_{l=1}^{N}\sum\limits_{j=1}^{N}|e_{kjl}|g_j^\natural g_l^\natural}{b_k}.$$

Then system (1) has a unique globally exponentially stable equilibrium in $\Sigma_{\widetilde{\mathcal{N}}}$.

Proof. The proof is similar to [9], we omit it here.

Remark 2. Our results can complement or improve the previous ones [7,8] and make the research method adopted in [5,6] and [10] more flexible for multistable analysis of neural networks.

4 Numerical Simulations

Consider the networks with two neurons:

$$\frac{du_k(t)}{dt}=-b_ku_k(t)+\sum_{j=1}^{2}w_{kj}g_j(u_j(t))+I_k$$

$$+\sum_{j=1}^{2}\sum_{l=1}^{2} e_{kjl} g_j\Big[u_j(t-\tau_{kj}(t))\Big] g_l\Big[u_l(t-\tau_{kl}(t))\Big],\ k=1,2. \qquad (7)$$

Simulation 1. We let $b_1 = 1$, $b_2 = 3$, $I_1 \equiv -4$, $I_2 \equiv -7.5$, $W = (w_{kj})_{2\times 2} = \begin{pmatrix} 8 & 0.5 \\ 0.1 & 15 \end{pmatrix}$, $E^{(1)} = (e_{1kj}) = 0$, $E^{(2)} = (e_{2kj}) = 0$, $\tau_{kj}(t) \equiv 0$, $g_j(\upsilon) := \frac{1}{1+e^{-\upsilon}}$.

Obviously, (H_1)-(H_2) hold and existing results in [3,4] are not applicable. Let $F_1(\upsilon) = -\upsilon + 8g_1(\upsilon)$ and $F_2(\upsilon) = -3\upsilon + 15g_2(\upsilon)$. We can check that $(H^D_{g_1})$, $(H^D_{g_2})$ hold and $p_1 \approx -1.763$, $q_1 \approx 1.763$, $p_2 \approx -0.962$, $q_2 \approx 0.962$.There exist 4 divisions $\widetilde{\mathcal{N}}_1 = (\emptyset, \emptyset, \{1,2\})$, $\widetilde{\mathcal{N}}_2 = (\{1\}, \emptyset, \{2\})$, $\widetilde{\mathcal{N}}_3 = (\{1,2\}, \emptyset, \emptyset)$, $\widetilde{\mathcal{N}}_4 = (\{2\}, \emptyset, \{1\})$ of $\mathcal{N} = \{1,2\}$ satisfying (3). From Theorem 1, we can check that system (7) has an equilibrium in each division region $M_l := \Sigma_{\widetilde{\mathcal{N}}_l}$ which is locally exponentially attractive $(l = 1,2,3,4)$. Fig. 1 shows the four division regions and the phase view of four equilibria of system (7) without high-order interconnections.

Simulation 2. Let $b_1 = 1$, $b_2 = 3$, $I_1 \equiv -4$, $I_2 \equiv -5$, $W = (w_{kj})_{2\times 2} = \begin{pmatrix} 8 & 0.1 \\ 0.1 & 15 \end{pmatrix}$, $E^{(1)} = (e_{1kj})_{2\times 2} = \begin{pmatrix} 0.1 & 0.1 \\ 0 & 0.1 \end{pmatrix}$, $E^{(2)} = (e_{2kj})_{2\times 2} = \begin{pmatrix} 2 & 0 \\ 0.1 & 0.5 \end{pmatrix}$, $g_j(\upsilon) \equiv g(\upsilon) := \frac{1}{1+e^{-\upsilon}}$, $\tau_{kj}(t) \equiv 10$, $k, j = 1,2$.

From Theorem 1, there exist only two locally exponentially attractive equilibria of system (7). For the phase view of system (7) with high-order interconnections, we can refer to Fig. 2.

Simulation 3. Let

$$b_1 = 4, \quad b_2 = 8, \quad I_1 \equiv 3, \quad I_2 \equiv -2, \quad W = (w_{kj})_{2\times 2} = \begin{pmatrix} 2 & 0 \\ 0 & 4 \end{pmatrix}, \ \tau_{kj}(t) \equiv 10$$

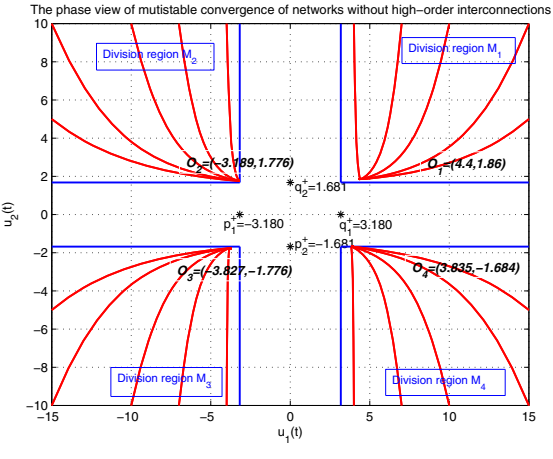


Fig. 1. The phase view of system (7) without high-order interconnections

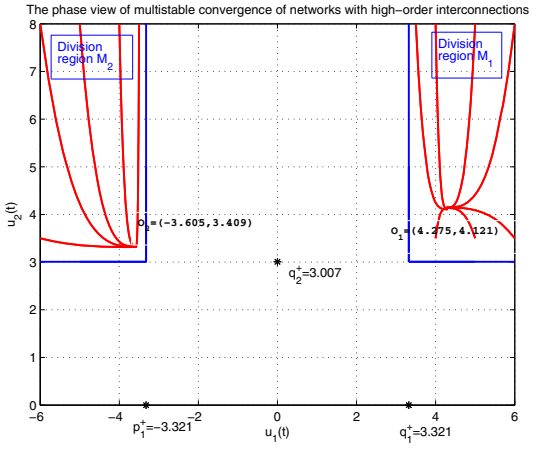


Fig. 2. The phase view of system (7) with high-order interconnections

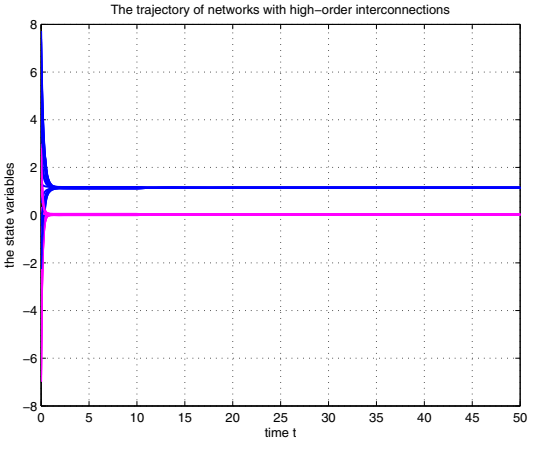


Fig. 3. The trajectory versus time of system (7) with high-order interconnections

$$E^{(1)} = \begin{pmatrix} 0.01 & 0.1 \\ 0 & 0.1 \end{pmatrix}, \quad E^{(2)} = \begin{pmatrix} 0.2 & 0 \\ 0.1 & 0.1 \end{pmatrix}, g_j(v) \equiv g(v) := \frac{e^v - e^{-v}}{e^v + e^{-v}}, \ k, j = 1, 2.$$

We can check that (H_1)-(H_2), $(H_{g_1}^U)$ and $(H_{g_2}^U)$ hold. Then there exists only one division $\widetilde{\mathcal{N}} = (\emptyset, \{1, 2\}, \emptyset)$ of $\mathcal{N} = \{1, 2\}$ satisfying (10). From Theorem 2, there exists a unique globally exponentially attractive equilibrium of system (7) in $M := \Sigma_{\widetilde{\mathcal{N}}}$. Fig. 3 shows the trajectory of state variable (u_1, u_2) of system (7).

Remark 3. Simulation 2 shows that the high-order synaptic connectivity impose additional load $\sum_{l=1}^{N}(|e_{kjl}|+|e_{klj}|)g_l^\natural$ (refer to (3)) to system (7) and hence system (7) has only two division regions which are invariant and locally attractive. The interesting phenomena show that high-order interactions of neurons may have less storage capacity than first-order ones.

5 Concluding Remarks

In this paper, we reveal a new result that high-order synaptic connectivity imposes an additional load to neural system and hence may reduce storage capacities; This conclusion refutes traditional viewpoint: high-order interactions of neurons have greater storage capacity than first-order ones.

Acknowledgments. This work was supported by the Foundation for Young Professors of Jimei University, the Scientific Research Foundation of Jimei University, the Foundation for Talented Youth with Innovation in Science and Technology of Fujian Province (2009J05009).

References

1. Cohen, M.A., Grossberg, S.: Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks. IEEE Trans. Syst. Man Cybern. SMC 13, 815–826 (1983)
2. Dembo, A., Farotimi, O., Kailath, T.: High-Order Absolutely Stable Neural Networks. IEEE Trans. Circuits Syst. 38, 57–65 (1991)
3. Xu, B., Liu, X., Liao, X.: Global Asymptotic Stability of High-Order Hopfield Type Neural Networks with Time Delays. Comput. Math. Appl. 45, 1729–1737 (2003)
4. Yu, Y., Cai, M.: Existence and Exponential Stability of Almost-Periodic Solutions for High-Order Hopfield Neural Networks. Math. Comput. Model. 47, 943–951 (2008)
5. Cheng, C., Lin, K., Shih, C.: Multistability in Recurrent Neural Networks. SIAM J. Appl. Math. 66, 1301–1320 (2006)
6. Cheng, C., Lin, K., Shih, C.: Multistability and Convergence in Delayed Neural Networks. Physica D 225, 61–74 (2007)
7. Huang, Z.K., Mohamad, S., Wang, W., Feng, C.: Convergence Analysis of General Neural Networks under Almost Periodic Stimuli. Int. J. Circ. Theor. Appl. 37, 723–750 (2009)
8. Huang, Z.K., Mohamad, S., Bin, H.: Multistability of HNNs with Almost Periodic Stimuli and Continuously Distributed Delays. Int. J. Syst. Sci. 40, 615–625 (2009)
9. Zeng, Z.G., Wang, J.: Multiperiodicity of Discrete-Time Delayed Neural Networks Evoked by Periodic External Inputs. IEEE Trans. Neural Netw. 17, 1141–1151 (2006)
10. Shih, C., Tseng, J.: Convergent Dynamics for Multistable Delayed Neural Networks. Nonlinearity 21, 2361–2389 (2008)

Properties of Periodic Solutions for Common Logistic Model with Discrete and Distributed Delay

Ting Zhang, Minghui Jiang, and Zhengwen Tu

Institute of Nonlinear and Complex System, China Three Gorges University, YiChang, Hubei 443002, China
zhangting0717@163.com

Abstract. This paper discusses about the existence and uniqueness of periodic solutions of Logistic model with discrete and distributed delay. By using a Lyapunov function which unlike the former method, the conditions of existence and uniqueness of periodic solutions for this Logistic model are obtained. The result of this paper extend results of periodic solutions for the delay differential equations.

Keywords: Logistic model, Lyapunov function, Periodic solutions.

1 Introduction

In this work, we consider a new general Logistic model with discrete and distributed delay, mainly discuss the existence and uniqueness of periodic solutions of this system:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = x(t)[r(t) - a(t)x(t-\tau) - b(t)\int_0^{+\infty} k(s)G(x(t-s))\mathrm{d}s] \tag{1}$$

with initial condition $x(t) = \phi(t), \phi(0) = 0$ for all $t \leq 0$, where $\phi(t) : (-\infty, 0] \to [0, +\infty)$ is continues and bounded with $\phi(0) = 0$. $k : [0, +\infty) \to [0, +\infty)$ is a piecewise continuous function and satisfies

$$\int_0^{+\infty} k(s)\mathrm{d}s = 1, \delta = \int_0^{+\infty} sk(s)\mathrm{d}s < +\infty \tag{2}$$

$r(t), a(t), b(t)$ are almost periodic on R, and for any $t \in R$ satisfy

$$0 < r_0 \leq r(t) \leq r_1 < \infty, 0 < a_0 \leq a(t) \leq a_1 < \infty, 0 < b_0 \leq b(t) \leq b_1 < \infty \tag{3}$$

$G(u)$ is a continuous function, and satisfies

$$G(0) = 0, G(u) \geq u > 0, 0 < l_0 \leq G'(u) \leq l_1 < \infty \tag{4}$$

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 635–642, 2010.

This work's results extend results of [1-4]. Set $G(u) = u$, then system (1) is reduced to the model (5) of [2]

$$N'(t) = N(t)[a(t) - c(t)N(t-\tau) - b(t)\int_0^{+\infty} k(s)N(t-s)\mathrm{d}s] \tag{5}$$

We lead the discrete delay τ into the model of [1], then system (1) can be obtained. Set $a(t) = 0, G(u) = u$, then system (1) is reduced to the model (6) of [3-4]

$$N'(t) = N(t)[a(t) - b(t)\int_0^{+\infty} k(s)N(t-s)\mathrm{d}s] \tag{6}$$

2 Preliminaries

In the [1,5], as we know, one can study the existence and uniqueness of periodic solutions for the function differential equations by using a Lyapunov function as Theorem A. in [1]. And we can get the following lemmas.

Lemma 2.1. Assume $x(t)$ is a solution of system (1) corresponding to the initial condition $\phi(t)$, then $x(t) > 0$ for all $t \geq 0$.

Proof. Assume when $0 \leq t \leq t_1$, $x(t_1) = 0$. Define $z(t)$ on $0 \leq t < t_1$: $z(t) = lnx(t)$, then we can obtained that $\lim_{t\to t_1^-} z(t) = -\infty$. Since $z'(t) = \frac{1}{x(t)}x'(t) = r(t) - a(t)e^{z(t-\tau)} - b(t)\int_0^{+\infty} k(s)G(e^{z(t-s)})\mathrm{d}s$, from (2)(3) and (4) we can obtain that $z'(t)$ bounded, then $z(t)$ bounded is obtained. This contradicts $\lim_{t\to t_1^-} z(t) = -\infty$. Then for any $t \geq 0$, $x(t) > 0$ is obtained.

Lemma 2.2. Assume that conditions (2)-(4) hold, for any solution $x(t)$ of system (1) corresponding to the initial condition $\phi(t)$ bounded for all $t \geq 0$.

Proof. Assume $x(t)$ is a solution of system (1), then follows Lemma 2.1. that $x(t) > 0$ for all $t \geq 0$. From (1), we have $x'(t) < x(t)r(t)$. It follows for $t \geq s > 0$ that $x(t) < x(t-s)exp[\int_{t-s}^{t} r(t)\mathrm{d}t] \leq x(t-s)e^{r_1 s}$, which implies that $x(t)e^{-r_1 s} < x(t-s), t \geq s > 0$, and $x(t)e^{-r_1\tau} < x(t-\tau), t \geq \tau$. Follows from (2) we have that $\forall\ B_0 > \varepsilon > 0, \forall\ t > T, \exists\ T > 0\ s.t.\ b_0 \int_t^{+\infty} k(s)e^{-r_1 s}\mathrm{d}s < \varepsilon$. Then from $t > T + \tau$ we get

$$x'(t) \leq x(t)[r(t) - a(t)x(t-\tau) - b(t)\int_0^{t} k(s)G(x(t-s))\mathrm{d}s]$$

$$< x(t)[r_1 - a_0 x(t)e^{-r_1\tau} - b_0 \int_0^t k(s)x(t)e^{-r_1 s}\mathrm{d}s]$$
$$< r_1 x(t) - [B_0 - \varepsilon]x^2(t),$$

where $B_0 = a_0 e^{-r_1\tau} + b_0 \int_t^{+\infty} k(s)e^{-r_1 s}\mathrm{d}s$.

Set $z(t) = \frac{1}{x(t)}$, then we have that $z'(t) > -r_1 z(t) + B_0 - \varepsilon$ for all $t > T + \tau$, so $z(t) > z(0)e^{-r_1\tau} + \frac{(B_0-\varepsilon)(1-e^{-r_1 t})}{r_1}$, for all $t > T+\tau$, which implies that $x(t) < \frac{x(0)r_1}{r_1 e^{-r_1 t}+(B_0-\varepsilon)(1-e^{-r_1 t})}$. Then we have that $\lim_{t\to+\infty} \sup x(t) \le \frac{r_1}{B_0-\varepsilon} < \frac{r_1}{B_0}$. This means that $x(t)$ is upper bounded for $t \ge 0$. So there exists a nonnegative constant β such that $0 < x(t) \le \frac{r_1}{B_0} + \beta$ for any $t \ge 0$.

Let $M(t) = \ln x(t)$, for any $t \ge 0$, system (1) can be written as

$$M'(t) = r(t) - a(t)e^{M(t-\tau)} - b(t) \int_0^{+\infty} k(s)G(e^{M(t-s)})\mathrm{d}s \tag{7}$$

From condition (2), we obtain $\delta = \int_0^{+\infty} sk(s)\mathrm{d}s = \int_0^{+\infty}\int_{t-s}^{t} k(s)\mathrm{d}v\mathrm{d}s < +\infty$. Let $M_1 = \ln(\frac{r_1}{B_0} + \beta)$, then $M(t) \le M_1$ for $t \ge 0$. And We have

$$M_2 = \sup_{t\ge 0} \int_0^{+\infty}\int_{t-s}^{t} b(v+s)k(s)G(e^{M(v)})\mathrm{d}v\mathrm{d}s < +\infty,$$
$$M_3 = \sup_{t\ge 0} \int_{t-\tau}^{t} a(s+\tau)e^{M(s)}\mathrm{d}s < +\infty.$$

Assume that there exists a constant $A = M_2 + M_3 > 0$ satisfies that for $M(t) < -A$, it have $r_0 - a_1 e^{M(t)} - b_1 G(e^{M(t)}) > 0$.

Consider a Lyapunov function as follows

$$V(t) = [M(t) - \int_{t-\tau}^{t} a(s+\tau)e^{M(s)}\mathrm{d}s - \int_0^{+\infty}\int_{t-s}^{t} b(v+s)k(s)G(e^{M(v)})\mathrm{d}v\mathrm{d}s]^2 \tag{8}$$

Let $B > \max\{(M_1+M_2+M_3)^2, (|M(0)|+M_2+M_3)^2, (A+M_2+M_3)^2\}$, then $V(0) < B$ and the derivative of $V(t)$ along the solution of system (8) is

$$V'_{(8)}(t) = 2[M(t) - \int_{t-\tau}^{t} a(s+\tau)e^{M(s)}\mathrm{d}s - \int_0^{+\infty}\int_{t-s}^{t} b(v+s)k(s)G(e^{M(v)})\mathrm{d}v\mathrm{d}s]$$
$$\times[r(t) - a(t+\tau)e^{M(t)} - G(e^{M(t)}) \int_0^{+\infty} b(t+s)k(s)\mathrm{d}s].$$

Let $\bar{t} = \sup\{t|0 \le t_1 \le t, V(t_1) \le B\}$, then $\bar{t} = +\infty$. Otherwise, if $\bar{t} < +\infty$, then there exists a σ such that $V(t_1) = B$, $V(t) > B$ for $\bar{t} < t < \bar{t} + \sigma$. Since $M(t) \le M_1(t \ge 0)$, then $\sup_{-A \le M(t) \le M_1}(M(t) - M_2 - M_3)^2 < B$, so we have $M(\bar{t}) < -A$. Since$V'_{(8)}(\bar{t}) < 0$, so $V(\bar{t}) < V(\bar{t}) = B$. This contradicts $V(t) > B(\bar{t} < t < \bar{t} + \sigma)$. So for any $t \ge 0$, we have $V(t) \le B$. From (7), there exists a constant m such that $M(t) > m$ for $t \ge 0$. It follows that $0 < m_1 \triangleq e^m \le x(t) \le \frac{r_1}{B_0} + \beta(t \ge 0)$.

3 The Existence and Uniqueness of Periodic Solution

Theorem 3.1. Assume that conditions (2)-(4) hold, and

$$r_1 b_1 \delta(a_1^2 + l_1^2 + 2b_1 l_1^2) + r_1 a_1 b_1 \tau(1 + l_1^2) + 2\tau r_1 a_1^2 \le 2B_0(a_0 + b_0 l_0) \tag{9}$$

$$\int_0^{+\infty} s^2 k(s) \mathrm{d}s < +\infty \tag{10}$$

Then there exists a unique periodic solution $\tilde{x}(t)$ of the system (1), and $0 < m_1 \le \tilde{x}(t) \le \frac{r_1}{B_0}$ for all $t \in R$, $\lim_{t \to +\infty}(x(t) - \tilde{x}(t)) = 0$, where $x(t)$ is an arbitrary solution of system (1).

Proof. From Lemma 2.1. and Lemma 2.2., we can know that

$$N'(t) = r(t) - a(t)e^{N(t-\tau)} - b(t) \int_0^{+\infty} k(s) G(e^{N(t-s)}) \mathrm{d}s] \tag{11}$$

$N(t) = \ln x(t), M(t) = \ln y(t)$ is the bounded solutions of system (11), and

$$0 < \ln m_1 \le N(t) \le \ln \frac{r_1}{B_0}, 0 < \ln m_1 \le M(t) \le \ln \frac{r_1}{B_0}, (t \in R)$$

$$\begin{cases} N'(t) = r(t) - a(t)e^{N(t-\tau)} - b(t) \int_0^{+\infty} k(s) G(e^{N(t-s)}) \mathrm{d}s \\ M'(t) = r(t) - a(t)e^{M(t-\tau)} - b(t) \int_0^{+\infty} k(s) G(e^{M(t-s)}) \mathrm{d}s \end{cases} \tag{12}$$

Now we construct a Lyapunov function as follows

$$V(t) = [N(t) - M(t) - \int_0^{+\infty} \int_{t-s}^{t} b(v+s)k(s)(G(e^{N(v)}) - G(e^{M(v)}))\mathrm{d}v\mathrm{d}s$$
$$- \int_{t-\tau}^{t} a(s+\tau)(e^{N(s)} - e^{M(s)})\mathrm{d}s]^2$$

$$+(a_1^2+a_1b_1)\int_{t-\tau}^{t}\int_{s}^{t}(e^{N(v)}-e^{M(v)})^2\mathrm{d}v\mathrm{d}s$$
$$+(b_1^2+b_1)\int_{0}^{+\infty}k(s)\int_{t-s}^{t}\int_{v}^{t}(G(e^{N(u)})-G(e^{M(u)}))^2\mathrm{d}u\mathrm{d}v\mathrm{d}s.$$

The derivative of $V(t)$ along the solution of system (12) is

$$\begin{aligned}V'_{(12)}(t)=&-2[N(t)-M(t)-\int_{t-\tau}^{t}a(s+\tau)(e^{N(s)}-e^{M(s)})\mathrm{d}s\\&-\int_{0}^{+\infty}\int_{t-s}^{t}b(v+s)k(s)(G(e^{N(v)})-G(e^{M(v)}))\mathrm{d}v\mathrm{d}s]\\&\times[a(t+\tau)(e^{N(t)}-e^{M(t)})+\int_{0}^{+\infty}b(t+s)k(s)(G(e^{N(t)})-G(e^{M(t)}))\mathrm{d}s]\\&+(b_1^2+b_1)\int_{0}^{+\infty}k(s)\int_{t-s}^{t}(G(e^{N(t)})-G(e^{M(t)}))^2rmdv\mathrm{d}s\\&-(b_1^2+b_1)\int_{0}^{+\infty}k(s)\int_{t-s}^{t}(G(e^{N(v)})-G(e^{M(v)}))^2\mathrm{d}v\mathrm{d}s\\&+(a_1^2+a_1b_1)\int_{t-\tau}^{t}(e^{N(t)}-e^{M(t)})^2\mathrm{d}s\\&-(a_1^2+a_1b_1)\int_{t-\tau}^{t}(e^{N(s)}-e^{M(s)})^2\mathrm{d}s\\=&-V_1(t)+V_2(t)+V_3(t)-V_4(t),\end{aligned}$$

where $V_1(t)=2(N(t)-M(t))[a(t+\tau)(e^{N(t)}-e^{M(t)})$
$$+\int_{0}^{+\infty}b(t+s)k(s)(G(e^{N(t)})-G(e^{M(t)}))\mathrm{d}s],$$

$$\begin{aligned}V_2(t)=2[&\int_{t-\tau}^{t}a(s+\tau)(e^{N(s)}-e^{M(s)})\mathrm{d}s\\&+\int_{0}^{+\infty}\int_{t-s}^{t}b(v+s)k(s)(G(e^{N(v)})-G(e^{M(v)}))\mathrm{d}v\mathrm{d}s]\end{aligned}$$

$$
\begin{aligned}
&\times[a(t+\tau)(e^{N(t)}-e^{M(t)})\\
&+\int_0^{+\infty} b(t+s)k(s)(G(e^{N(t)})-G(e^{M(t)}))\mathrm{d}s],\\
V_3(t) &= (b_1^2+b_1)\int_0^{+\infty} k(s)\int_{t-s}^{t}(G(e^{N(t)})-G(e^{M(t)}))^2\mathrm{d}v\mathrm{d}s\\
&+(a_1^2+a_1b_1)\int_{t-\tau}^{t}(e^{N(t)}-e^{M(t)})^2\mathrm{d}s\\
&= (b_1^2+b_1)(G(e^{N(t)})-G(e^{M(t)}))^2\int_0^{+\infty}k(s)\int_{t-s}^{t}\mathrm{d}v\mathrm{d}s\\
&+(a_1^2+a_1b_1)(e^{N(t)}-e^{M(t)})^2\int_{t-\tau}^{t}\mathrm{d}s\\
&= \delta(b_1^2+b_1)(G(e^{N(t)})-G(e^{M(t)}))^2+\tau(a_1^2+a_1b_1)(e^{N(t)})-e^{M(t)})^2,\\
V_4(t) &= (b_1^2+b_1)\int_0^{+\infty} k(s)\int_{t-s}^{t}(G(e^{N(v)})-G(e^{M(v)}))^2\mathrm{d}v\mathrm{d}s\\
&+(a_1^2+a_1b_1)\int_{t-\tau}^{t}(e^{N(s)}-e^{M(s)})^2\mathrm{d}s\\
\text{and } V_2(t) &\le \int_0^{+\infty}\int_{t-s}^{t} b(v+s)k(s)[(G(e^{N(v)})-G(e^{M(v)}))^2+a^2(t+\tau)\\
&(e^{N(t)}-e^{M(t)})^2]\mathrm{d}v\mathrm{d}s+\int_0^{+\infty}\int_{t-s}^{t} b(v+s)k(s)[(G(e^{N(v)})-G(e^{M(v)}))^2\\
&+(G(e^{N(t)})-G(e^{M(t)}))^2]\mathrm{d}v\mathrm{d}s\times\int_0^{+\infty} b(t+s)k(s)\mathrm{d}s\\
&+\int_{t-\tau}^{t} a(s+\tau)a(t+\tau)[(e^{N(s)}-e^{M(s)})^2+(e^{N(t)}-e^{M(t)})^2]\mathrm{d}s\\
&+\int_{t-\tau}^{t} a(s+\tau)[(e^{N(s)}-e^{M(s)})^2+(G(e^{N(t)})-G(e^{M(t)}))^2]\mathrm{d}s
\end{aligned}
$$

$$
\begin{aligned}
&\times \int_0^{+\infty} b(t+s)k(s)\mathrm{d}s \\
&\le b_1 a_1^2 \int_0^{+\infty} k(s) \int_{t-s}^{t} (e^{N(t)} - e^{M(t)})^2 \mathrm{d}v\mathrm{d}s \\
&\quad + b_1^2 \int_0^{+\infty} k(s) \int_{t-s}^{t} [(G(e^{N(t)}) - G(e^{M(t)}))^2] \mathrm{d}v\mathrm{d}s \\
&\quad + (b_1 + b_1^2) \int_0^{+\infty} k(s) \int_{t-s}^{t} [(G(e^{N(v)}) - G(e^{M(v)}))^2] \mathrm{d}v\mathrm{d}s \\
&\quad + a_1^2 \int_{t-\tau}^{t} (e^{N(s)} - e^{M(s)})^2 \mathrm{d}s + a_1 b_1 \int_{t-\tau}^{t} (e^{N(s)} - e^{M(s)})^2 \mathrm{d}s \\
&\quad + a_1^2 \int_{t-\tau}^{t} (e^{N(t)} - e^{M(t)})^2 \mathrm{d}s + a_1 b_1 \int_{t-\tau}^{t} (G(e^{N(t)}) - G(e^{M(t)}))^2 \mathrm{d}s \\
&= \delta b_1 a_1^2 (e^{N(t)} - e^{M(t)})^2 + \delta b_1^2 (G(e^{N(t)}) - G(e^{M(t)}))^2 \\
&\quad + \tau a_1^2 (e^{N(t)} - e^{M(t)})^2 + \tau a_1 b_1 (G(e^{N(t)}) - G(e^{M(t)}))^2 + V_4(t).
\end{aligned}
$$

Using the Mean Value Theorem, we can get that

$$
(N(t) - M(t))(e^{N(t)} - e^{M(t)}) = \frac{(e^{N(t)} - e^{M(t)})^2}{e^{\theta(t)N(t) + (1-\theta(t))M(t)}} \ge \frac{B_0 (e^{N(t)} - e^{M(t)})^2}{r_1}.
$$

So we have

$$
\begin{aligned}
V'_{(12)}(t) &\le -2(N(t) - M(t))[a(t+\tau)(e^{N(t)} - e^{M(t)}) \\
&\quad + \int_0^{+\infty} b(t+s)k(s)(G(e^{N(t)}) - G(e^{M(t)}))\mathrm{d}s] + \delta b_1 a_1^2 (e^{N(t)} - e^{M(t)})^2 \\
&\quad + \delta b_1^2 (G(e^{N(t)}) - G(e^{M(t)}))^2 + \tau a_1^2 (e^{N(t)} - e^{M(t)})^2 \\
&\quad + \tau a_1 b_1 (G(e^{N(t)}) - G(e^{M(t)}))^2 + \delta(b_1^2 + b_1)(G(e^{N(t)}) - G(e^{M(t)}))^2 \\
&\quad + \tau(a_1^2 + a_1 b_1)(e^{N(t)} - e^{M(t)})^2 + V_4(t) - V_4(t) \\
&\le -2a_0 \frac{B_0 (e^{N(t)} - e^{M(t)})^2}{r_1} - 2b_0 (N(t) - M(t)) G'(\xi)(e^{N(t)} - e^{M(t)}) \\
&\quad + (\delta b_1 a_1^2 + \tau a_1^2 + \tau a_1^2 + \tau a_1 b_1)(e^{N(t)} - e^{M(t)})^2 \\
&\quad + (\tau a_1 b_1 + \delta b_1^2 + \delta b_1^2 + \delta b_1)(G(e^{N(t)}) - G(e^{M(t)}))^2 \\
&\le -[\frac{2B_0(a_0 + b_0 l_0)}{r_1} - \delta b_1 (a_1^2 + l_1^2 + 2b_1 l_1^2) \\
&\quad - \tau a_1 b_1 (1 + l_1^2) - 2\tau a_1^2](e^{N(t)} - e^{M(t)})^2.
\end{aligned}
$$

Set $r = \frac{2B_0(a_0+b_0l_0)}{r_1} - \delta b_1(a_1^2 + l_1^2 + 2b_1l_1^2) - \tau a_1 b_1(1 + l_1^2) - 2\tau a_1^2)$, From (10), we can know that $r > 0$, it follows that $V'_{(12)}(t) \leq -r(e^{N(t)} - e^{M(t)})^2$, using the Mean Value Theorem again, we can get $V'_{(12)}(t) \leq -rm_1^2(N(t) - M(t))^2, t \in R$. It means that there exists a positive constant c such that

$$V'_{(12)}(t) \leq -cV(t), t \in R. \tag{13}$$

And we can easily know that $V(t)$ satisfies the conditions (1)(2) of Theorem A. in [1], so from Theorem A. in [1], we can get that there exists a unique positive periodic solution $\tilde{N}(t)$ of system (11), it follows that there exists a unique positive periodic solution $\tilde{x}(t)$ of system (1), and $0 < m_1 \leq \tilde{x}(t) \leq \frac{r_1}{B_0}, t \in R$. From (13) we know that $N(t), M(t)$ is the solution of system (11), and $\lim_{t\to+\infty}(N(t) - M(t)) = 0$. This means that $\lim_{t\to+\infty}(x(t) - \tilde{x}(t)) = 0$.

Remark 1. The Theorem 3.1. completely answered the proposal of [5].

Remark 2. This work's results improved the results of [1-4].

4 Conclusion

This paper has mainly studied the problem about the existence and uniqueness of periodic solutions of Logistic model with discrete and distributed delay. By emplying Lyapunov methods, we get the more general conditions of the properties of periodic solutions in the Logistic model. The results improved some known results of periodic solutions for the delay differential equations.

Acknowledgments

This work is supported by the Graduate Scientific Research Creative Foundation of Three Gorges University (200945) (200946) and the Scientific Innovation TeamProject of Hubei Provincial Department of Education (T200809), the Doctoral Start-up Funds of China Three Gorges University(0620060061) and Hubei Natural Science Foundation (No.2008CDZ046).

References

1. Feng, C.: Almost periodic solutions for some integrodifferential equations with infinite delay. Applied Mathematics Letters 18, 245–252 (2005)
2. Meng, X.Z., Chen, L.S., Wang, X.L.: Some new results for a logistic almost periodic system with infinite delay and discrete delay. Real World Applications 10, 1255–1264 (2009)
3. Feng, C.: On the existence and uniqueness of almost periodic solutions for delay Logistic equations. Applied Mathematics Letters 136, 487–494 (2005)
4. Li, H.X.: Almost periodic solutions for logistic equations with infinite delay. Applied Mathematics Letters 21, 113–118 (2008)
5. Seifert, G.: Almost periodic solutions for delay Logistic equations with almost periodic time dependence. Differential Integral Equations 9(2), 335–342 (1996)
6. Sawano, K.: Exponential asymptotic stability for functional differential equations with infinite retardation. Tohoku Math. J. 31, 363–382 (1979)

New Results of Globally Exponentially Attractive Set and Synchronization Controlling of the Qi Chaotic System

Jigui Jian, Xiaolian Deng, and Zhengwen Tu

Institute of Nonlinear and Complex Systems, China Three Gorges University, Yichang, Hubei, 443002, China
jiguijian65@hotmail.com

Abstract. This paper treats the globally exponentially attractive set and synchronization problem of the Qi chaotic system. Firstly, based on the generalized Lyapunov function theory, a new ellipsoid estimation of the globally exponentially attractive set and positive invariant set of the Qi chaotic system was given without existence assumptions. Secondly, based on some inequalities techniques and matrix theory, nonlinear feedback control with two inputs was used to realize the globally exponentially synchronization of two chaotic systems. Some sufficient algebraic criteria for the globally exponential synchronization of two chaotic systems are obtained analytically. Finally, numerical simulations are presented to show the effectiveness of the proposed chaos synchronization scheme.

Keywords: Chaos, Globally exponentially attractive set, Positive invariant set, Generalized Lyapunov function, Feedback control.

1 Introduction

Since the discovery of the Lorenz chaotic attractor in 1963, many other chaotic systems have been found, including the well-known R*ö*ssler system, Chua's circuit, Chen and L*ü* systems, which have been served as models for study of chaos. Because of sensitive dependence on initial conditions, chaotic systems are difficult to be synchronized or controlled. From the earlier works, the researchers have realized that synchronization of chaotic motions are possible, synchronization of chaos was of great interest in these years[1-7].

Recently, estimating the bound of chaotic systems has been a flurry of research activities. Because the ultimate boundedness of a chaotic system is very important for the study of the qualitative behavior of a chaotic system. If one can show that a chaotic system under consideration has a globally attractive set, then one knows that the system cannot have equilibrium points, periodic or quasi-periodic solutions, or other chaotic attractors existing outside the attractive set. This greatly simplifies the analysis of dynamics of the system. The ultimate boundedness also plays a very important role in the designs of chaos control

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 643–650, 2010.

and chaos synchronization[2-7]. For the Lorenz system, the ultimate boundedness property was first studied by G. Leonov and some important results were obtained[8]. Since then, the ultimate boundedness property of the Lorenz system has further been investigated by many researchers [3, 9-14] and some more explicit estimates of the globally attractive set and the positive invariant set for the Lorenz system are provided. The bound and globally attractive set of the Chen system were also investigated by [15]. However, so far very little has been achieved on the other chaotic systems [2, 4-7], regarding the property of ultimate boundedness.

In paper [16], a new chaotic system named the Qi system is set up, which is a third order autonomous system exhibiting very complex dynamical behaviors with some interesting characteristics. For the Qi chaotic system, there were few results for its globally attractive set[2, 4], where some ellipsoidal estimates of the globally attractive set and the positive invariant set were obtained, but the result [2] for all positive parameters of the system have not been considered. In this paper, we will further investigate the ultimate bound and positive invariant set for the Qi chaotic system by the generalized Lyapunov function theory and present some new formula to estimate the globally exponentially attractive set and positive invariant set. Meanwhile, nonlinear feedback control with two inputs was used to realize the globally exponentially synchronization of two chaotic systems and new sufficient algebraic criteria for the globally exponential synchronization of two chaotic systems are obtained analytically.

Chaotic system can be described by the following system of differential equations [16]

$$\begin{cases} \dot{x} = a(y - x) + yz, \\ \dot{y} = cx - y - xz, \\ \dot{z} = xy - bz, \end{cases} \tag{1}$$

where a, b and c are parameters that are all positive real. When the system parameters $a = 35, b = \frac{8}{3}$ and $c \in (17, 189)$, this system exhibits chaotic behavior [16] and see Figure 1. Without the particular statement, these values are adopted in this whole paper.

Definition 1. [10] For generalized radially unbound and positive definite function $V_\lambda(X) = V_\lambda(x, y, z)$ with $\lambda \geq 0$, if there exists a constant number $L_\lambda > 0$ such that for $V_\lambda(X_0) > L_\lambda$ and $V_\lambda(X) > L_\lambda$ imply $\lim_{t\to+\infty} V_\lambda(X(t)) = L_\lambda$, then $\Omega_\lambda = \{X|V_\lambda(X(t) \leq L_\lambda)\}$ is said to be a globally attractive set of the system (1). If for any $X_0 \in \Omega_\lambda$ and any $t \geq t_0$ imply $X(t, t_0, X_0) \in \Omega_\lambda$, then Ω_λ is said to be positive invariant set. If there exists constant numbers $L_\lambda > 0$, $r_\lambda > 0$ and $\forall X_0 \in R^3$ such that for $V_\lambda(X_0) > L_\lambda$ and $V_\lambda(X(t)) > L_\lambda$ imply $V_\lambda(X(t)) - L_\lambda \leq (V_\lambda(X_0) - L_\lambda)e^{-r_\lambda(t-t_0)}$, then $\Omega_\lambda = \{X|V_\lambda(X(t) \leq L_\lambda)\}$ is said to be a globally exponentially attractive set of the system (1). Where $X = X(t) = (x(t), y(t), z(t))^T, X_0 = X(t_0)$.

2 Globally Exponentially Attractive Set and Positive Invariant Set of the Qi Chaotic System

Theorem 1. Suppose $a > 0, b > 0$ and $c > 0$, for any constant $\lambda > 0$, then the ellipsoid

$$\Omega_\lambda : \lambda x^2 + (\lambda+2)y^2 + 2(z - \frac{\lambda a + (\lambda+2)c}{2})^2 \le R_\lambda^2, \tag{2}$$

is a globally attractive set and positive invariant set of the system (1). Where

$$R_\lambda^2 = \begin{cases} \frac{(\lambda a+(\lambda+2)c)^2}{2}, 0 < b \le 1 \le a; \\ \frac{b^2}{8(b-1)}(\lambda a + (\lambda+2)c)^2, a \ge 1, b > 1; \\ \frac{b^2}{8a(b-a)}(\lambda a + (\lambda+2)c)^2, 0 < a < b \le 1 \ or \ a < 1 < b. \end{cases}$$

Proof. Consider the following generalized positive definite Lyapunov function with radially unbound

$$V_\lambda = \frac{1}{2}\lambda x^2 + \frac{1}{2}(\lambda+2)y^2 + (z - \frac{\lambda a + (\lambda+2)c}{2})^2. \tag{3}$$

At first, we consider the case of $0 < b \le 1$. Employing Lyapunov function (2) and computing the derivative of V_λ along the positive half-trajectory of the system (1), we have

$$\frac{dV_\lambda}{dt}|_{(1)} = -\lambda a x^2 - (\lambda+2)y^2 - 2bz^2 + b(\lambda a + (\lambda+2)c)z. \tag{4}$$

For $a \ge 1$, we obtain

$$\begin{aligned} \tfrac{dV_\lambda}{dt}|_{(1)} &\le -\tfrac{b}{2}\lambda x^2 - \tfrac{b}{2}(\lambda+2)y^2 - bz^2 + b(\lambda a + (\lambda+2)c)z \\ &= -\tfrac{b}{2}(\lambda x^2 + (\lambda+2)y^2 + 2(z - \tfrac{\lambda a+(\lambda+2)c}{2})^2) + \tfrac{b}{4}(\lambda a + (\lambda+2)c)^2 \\ &= -bV_\lambda(X(t)) + \tfrac{b}{4}(\lambda a + (\lambda+2)c)^2 < 0 \end{aligned}$$

for $\lambda x^2 + (\lambda+2)y^2 + 2(z - \frac{\lambda a+(\lambda+2)c}{2})^2 > \frac{1}{2}(\lambda a + (\lambda+2)c)^2$ and

$$V_\lambda(X(t)) - \frac{1}{4}(\lambda a + (\lambda+2)c)^2 \le (V_\lambda(X(t_0)) - \frac{1}{4}(\lambda a + (\lambda+2)c)^2)e^{-b(t-t_0)}.$$

Then, the ellipsoid $\lambda x^2 + (\lambda+2)y^2 + 2(z - \frac{\lambda a+(\lambda+2)c}{2})^2 \le \frac{1}{2}(\lambda a + (\lambda+2)c)^2$ for $a \ge 1$ and $0 < b \le 1$ is the globally exponentially attractive set and positive invariant set of the system (1).

For $a \ge 1$ and $b > 1$, let

$$F_2(X) = \lambda(1-a)x^2 - 2(b-1)(z - \frac{(b-2)(\lambda a + (\lambda+2)c)}{4(b-1)})^2 + \frac{b^2(\lambda a + (\lambda+2)c)^2}{8(b-1)},$$

then $F_2(X) \le \frac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-1)}$. We compute the derivative of V_λ along the trajectory of the system (1) as follows

$$\begin{aligned} \tfrac{dV_\lambda}{dt}|_{(1)} &= -\lambda x^2 - (\lambda+2)y^2 - 2(z - \tfrac{\lambda a+(\lambda+2)c}{2})^2 + F_2(X) \\ &\le -2V_\lambda(X(t)) + \tfrac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-1)} < 0 \end{aligned}$$

for $\lambda x^2 + (\lambda+2)y^2 + 2(z - \frac{\lambda a+(\lambda+2)c}{2})^2 > \frac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-1)}$, and

$$V_\lambda(X(t)) - \frac{b^2(\lambda a+(\lambda+2)c)^2}{16(b-1)} \leq (V_\lambda(X(t_0)) - \frac{b^2(\lambda a+(\lambda+2)c)^2}{16(b-1)})e^{-2(t-t_0)}.$$

Then, the ellipsoid $\lambda x^2+(\lambda+2)y^2+2(z-\frac{\lambda a+(\lambda+2)c}{2})^2 \leq \frac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-1)}$ for $b>1$ and $a \geq 1$ is the globally exponentially attractive set and positive invariant set of the system (1).

For $0 < a < b \leq 1$ or $a < 1 < b$, let

$$F_1(X) = (a-1)(\lambda+2)y^2 - 2(b-a)(z - \frac{(b-2a)(\lambda a+(\lambda+2)c)}{4(b-a)})^2 + \frac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-a)},$$

then $F_1(X) \leq \frac{b^2(\lambda a+(\lambda+2)c))^2}{8(b-a)}$. From (3) and (4), we have

$$\begin{aligned} \frac{dV_\lambda}{dt}|_{(1)} &= -a(\lambda x^2 + (\lambda+2)y^2 + 2(z - \tfrac{\lambda a+(\lambda+2)c}{2})^2) + F_1(X) \\ &\leq -2aV_\lambda(X(t)) + \tfrac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-a)} < 0 \end{aligned}$$

for $\lambda x^2 + (\lambda+2)y^2 + 2(z - \frac{\lambda a+(\lambda+2)c}{2})^2 > \frac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-a)}$, and

$$V_\lambda(X(t)) - \frac{b^2(\lambda a+(\lambda+2)c)^2}{16a(b-a)} \leq (V_\lambda(X(t_0)) - \frac{b^2(\lambda a+(\lambda+2)c)^2}{16a(b-a)})e^{-2a(t-t_0)}.$$

So, the ellipsoid $\lambda x^2 + (\lambda+2)y^2 + 2(z - \frac{\lambda a+(\lambda+2)c}{2})^2 \leq \frac{b^2(\lambda a+(\lambda+2)c)^2}{8(b-a)}$ for $0 < a < b \leq 1$ or $a < 1 < b$ is the globally exponentially attractive set and positive invariant set of the system (1). This proof is completed.

3 The Application in Chaos Synchronization

We assume that we have two systems and that the drive system with the subscript 1 is to control the response system with subscript 2, then the drive system (5) and the response system (6) are defined as follows, respectively

$$\begin{cases} \dot{x_1} = a(y_1 - x_1) + y_1 z_1, \\ \dot{y_1} = cx_1 - y_1 - x_1 z_1, \\ \dot{z_1} = x_1 y_1 - b z_1, \end{cases} \tag{5}$$

and

$$\begin{cases} \dot{x_2} = a(y_2 - x_2) + y_2 z_2 + u_1(t), \\ \dot{y_2} = cx_2 - y_2 - x_2 z_2 + u_2(t), \\ \dot{z_2} = x_2 y_2 - b z_2 + u_3(t). \end{cases} \tag{6}$$

We define the error system as the differences between the systems (5) and (6) by using $e(t) = (e_1, e_2, e_3)^T = (x_2 - x_1, y_2 - y_1, z_2 - z_1)^T$. Using this notation, we get an error dynamic system as

$$\begin{cases} \dot{x_1} = -ae_1 + ae_2 + e_2 e_3 + y_1 e_3 + z_1 e_2 + u_1(t), \\ \dot{y_1} = ce_1 - e_2 - e_1 e_3 - z_1 e_1 - x_1 e_3 + u_2(t), \\ \dot{z_1} = -be_3 + e_1 e_2 + x_1 e_2 + y_1 e_1 + u_3(t). \end{cases} \tag{7}$$

Theorem 2. Suppose that M_y and M_z defined by (8) and (9) from (2) are the upper bounds of the absolute values of variables y_1 and z_1, respectively. Then the origin of the system (7) is exponentially stable with one of the following control laws

(A_1) $u_1 = -ke_1, u_2 = \frac{1}{2}x_1e_3, u_3 = 0;$
(A_2) $u_1 = -ke_1, u_2 = 0, u_3 = x_1e_2.$
Where $k > \frac{M_y^2}{b} + \frac{(a+2c+M_z)^2}{8} - a$, and

$$M_y = \begin{cases} \frac{a+3c}{\sqrt{6}}, & 0 < b \le 1 \le a; \\ \frac{b(a+3c)}{2\sqrt{2(b-1)}}, & a \ge 1, b > 1; \\ \frac{b(a+3c)}{2\sqrt{2a(b-a)}}, & 0 < a < b \le 1 \ or \ a < 1 < b. \end{cases} \tag{8}$$

$$M_z = \begin{cases} a + 3c, & 0 < b \le 1 \le a; \\ \frac{b+2\sqrt{b-1}}{4\sqrt{b-1}}(a+3c), & a \ge 1, b > 1; \\ \frac{b+2\sqrt{a(b-a)}}{4\sqrt{a(b-a)}}(a+3c), & 0 < a < b \le 1 \ or \ a < 1 < b. \end{cases} \tag{9}$$

Proof. Construct a Lyapunov function in the form of $V = \frac{1}{2}e_1^2 + e_2^2 + \frac{1}{2}e_3^2$, then

$$\begin{aligned} \dot{V}|_{(7)} &= -ae_1^2 - 2e_2^2 - be_3^2 + (a + 2c - z_1)e_1e_2 + 2y_1e_1e_3 - x_1e_2e_3 \\ &\quad + e_1u_1 + 2e_2u_2 + e_3u_3 \\ &= -(a+k)e_1^2 - 2e_2^2 - be_3^2 + (a + 2c - z_1)e_1e_2 + 2y_1e_1e_3 \\ &= -(e_1, e_2, e_3)Q(e_1, e_2, e_3)^T, \end{aligned}$$

where $Q = \begin{pmatrix} a+k & \frac{z_1-a-2c}{2} & -y_1 \\ \frac{z_1-a-2c}{2} & 2 & 0 \\ -y_1 & 0 & b \end{pmatrix}$. Obviously, to ensure that the origin of the error system (7) is exponentially stable, we let the matrix Q be positive definite. This is the case if and only if the following three inequalities hold:

$$k + a > 0, 8b(k + a) > (z_1 - a - 2c)^2, 8b(k + a) > 8y_1^2 + b(z_1 - a - 2c)^2. \tag{10}$$

It is easy to show that the condition (10) holds if

$$8b(k + a) > 8M_y^2 + b(a + 2c + M_z)^2$$

is satisfied. Then the matrix Q is positive definite, and $\dot{V}|_{(7)}$ is negative definite, we have

$$\dot{V}|_{(7)} \le -\lambda_{min}(Q)V,$$

i.e.

$$e_1^2 + e_2^2 + e_3^2 \le 2(e_1^2(t_0) + e_2^2(t_0) + e_3^2(t_0))e^{-\lambda_{min}(Q)(t-t_0)}. \tag{11}$$

The inequality (11) implies that the origin of the error system (7) is exponentially asymptotically stable. Therefore, the drive system (5) is exponentially synchronizing with the response system (6). This concludes the proof.

Corollary 1. Let $k > 0$ and $8(k+a) > (a+2c+2M_z)^2$ with M_z defined by (9), then the origin of the error system (7) is exponentially stable, and consequently, the two chaotic systems (5) and (6) can be exponentially synchronized with one of the following control laws:

(B_1) $u_1 = -ke_1 - 2y_1e_3, u_2 = \frac{1}{2}x_1e_3, u_3 = 0;$
(B_2) $u_1 = -ke_1, u_2 = 0, u_3 = x_1e_2 - 2y_1e_1;$
(B_3) $u_1 = -ke_1 - 2y_1e_3, u_2 = 0, u_3 = x_1e_2.$

Corollary 2. Let $k > 0$ and $8(k+a) > (a+2c)^2$, then the origin of the error system (7) is exponentially stable, and consequently, the two chaotic systems (5) and (6) can be exponentially synchronized with one of the following control laws:

(C_1) $u_1 = -ke_1 + z_1e_2, u_2 = 0, u_3 = x_1e_2 - 2y_1e_1;$
(C_2) $u_1 = -ke_1 + z_1e_2 - 2y_1e_3, u_2 = \frac{1}{2}x_1e_3, u_3 = 0.$

Remark 1. The condition $8(k+a) > (a+2c)^2$ in Corollary 2 gives an estimate for the range of the control gain k. Specifically, we can determine a critical value $k^* = \frac{1}{8}(a+2c)^2 - a > 0$ such that complete synchronization can be achieved if $k > k^*$. In fact, if let $a = 35, c = 18$, then it is easy to calculate $k^* = 595.125$.

4 Numerical Simulation

To verify the theoretical results given in the previous sections, we will discuss the simulation results for the chaotic systems with the parameters $a = 35, b = \frac{8}{3}, c = 18$. The initial conditions of drive and response systems are $(2, 3, 4)$ and $(0, 6, -1)$, respectively. In order to choose the control parameters in Theorem 2, $M_y \geq |y|$ and $M_z \geq |z|$ need to be estimated. Through simulations, we obtain $M_x = 23.3540, M_y = 16.6188, M_z = 30.5374$. The controllers (A_1) with $k = 1340$ is chosen as the control law for the system (7), then the response system (6) synchronizes with the drive system (5) as shown in Fig.2. Fig.3(a) and Fig.3(b) show the trajectory e_1, e_2, e_3 of the error system (7) with the control laws (B_1) for $k = 1240$ and (C_1) for $k = 600$, respectively.

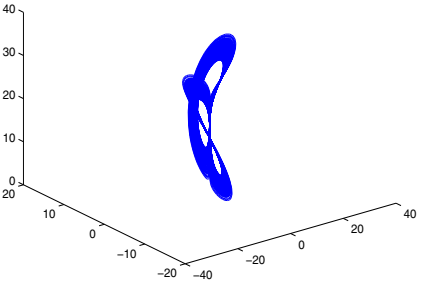


Fig. 1. System (1) exhibits chaotic behavior

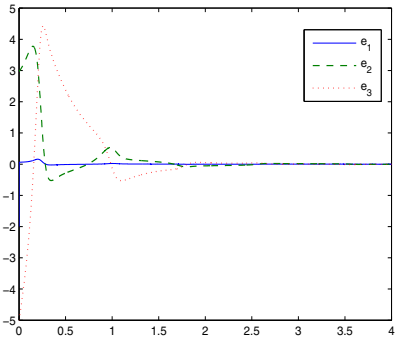


Fig. 2. Shows synchronization errors with the control (A_1)

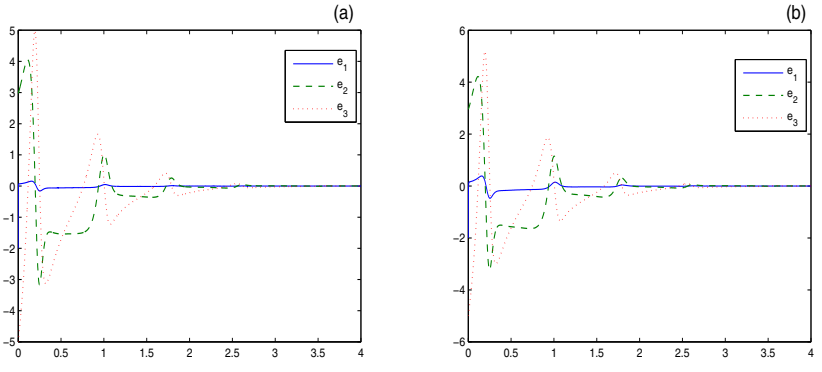


Fig. 3. (a) shows synchronization errors with the control (B_2); (b) shows synchronization errors with the control (C_1)

.

5 Conclusion

According to the parameters, a new detailed ellipsoid estimation of exponential attractive sets for the Qi chaotic system is presented without any hypothesis on the existence in this paper. Meanwhile, nonlinear feedback control with two inputs is investigated and some sufficient conditions for the globally exponential synchronization of two chaotic systems are obtained analytically. All the numerical simulation results are in line with the theoretical analysis.

Acknowledgments

This work was partially supported by the Scientific Innovation Team Project (T200809) and the Scientific Research Projects (D20091305) of Hubei Provincial Department of Education, the National Natural Science Foundation of China (60974136) and the Hubei Provincial Natural Science Foundation (2008CDB316,

2008CDZ046) and the Doctoral Pre-Research Foundation of China Three Gorges University.

References

1. Mei, X.H., Yu, J.N., Zhang, J.G.: The linear and nonlinear feedback synchronization of a class of chaotic finance system. Journal of Tianjin Normal University (Natural Science Edition) 28(3), 49–51 (2008)
2. Fu, W.F., Jiang, M.H.: Globally Exponentially Attractive Set and Feedback Synchronization of a New Chaos System. Journal of Wuhan University of Technology 30(7), 103–106 (2008)
3. Liao, X.X., Fu, Y.L., Xie, S.L.: On the new results of global attractive set and positive invariant set of the Lorenz system and the applications to the chaos control and synchronization. Science in China Series F 48(3), 304–321 (2005)
4. Xu, H.X., Shu, Y.L., Yuan, G.X.: The ultimate bound of a chaotic system and its application in chaos synchronization. J. Chongqing Technol. Business Univ. (Nat. Sci. Ed.) 25(6), 564–568 (2008)
5. Jian, J.G., Deng, X.L., Wang, J.F.: Globally Exponentially Attractive Set and Synchronization of a Class of Chaotic Finance System. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009, Part I. LNCS, vol. 5551, pp. 253–261. Springer, Heidelberg (2009)
6. Jian, J.G., Tu, Z.W., Yu, H.: Estimating the Globally Attractive Set and Positively Invariant Set of a New Lorenz-like Chaotic System and Its Applications. In: 2009 International Workshop on Chaos-Fractals Theories and Applications, pp. 241–245 (2009)
7. Shu, Y.L., Xu, H.X., Zhao, Y.H.: Estimating the ultimate bound and positively invariant set for a newchaotic system and its application in chaos synchronization. Chaos, Solitons and Fractals (2009), doi:10.1016/j.chaos.2009.04.003
8. Leonov, G., Bunin, A., Koksch, N.: Attract or localization of the Lorenz system. ZAMM 67(2), 649–656 (1987)
9. Liao, X.X., Fu, Y.L., Xie, S.L.: Globally exponentially attractive sets of the family of Lorenz systems. Science in China Series F 51(3), 283–292 (2008)
10. Liao, X.X., Luo, H.G., Fu, Y.L., Xie, S.L., Yu, P.: Positive Invariant Set and the Globally Exponentially Attractive Set of Lorenz System Group. Science in China-E 37(6), 757–769 (2007)
11. Li, D.M., Lu, J.A., Wu, X.Q., et al.: Estimating the ultimate bound and positive invariant set for the Lorenz system and a unified chaotic system. Journal of Mathematical Analysis and Application 323(2), 844–853 (2006)
12. Yu, P., Liao, X.X., Xie, S.L., Fu, Y.L.: A constructive proof on the existence of globally exponentially attractive set and positive invariant set of general Lorenz family. Commun. Nonlinear Sci. Numer. Simulat. 14, 2886–2896 (2009)
13. Shu, Y.L., Zhang, Y.H.: Estimating the ultimate bound and positive invariant set for a generalized Lorenz system. Journal of Chongqing University (English Edition) 7(2), 151–154 (2008)
14. Sun, Y.J.: Solution bounds of generalized Lorenz chaotic systems. Chaos, Solitons and Fractals 40, 691–696 (2009)
15. Qin, W.X., Chen, G.R.: On the boundedness of solutions of the Chen system. Journal of Mathematical Analysis and Application 329(1), 445–451 (2007)
16. Qi, G.Y., Chen, G.R., Du, S.Z.: Analysis of a new chaotic system. Physica A 352, 295–308 (2005)

Stability and Attractive Basin of Delayed Cohen-Grossberg Neural Networks

Ailong Wu[1], Chaojin Fu[1], and Xian Fu[2]

[1] College of Mathematics and Statistics, Hubei Normal University, Huangshi 435002, China
[2] College of Computer Science and Technology, Hubei Normal University, Huangshi 435002, China
alequ@126.com

Abstract. This paper addresses the local stability of the delayed Cohen-Grossberg neural networks (CGNNs). A sufficient condition for the local exponential stability of an equilibrium point is presented, and the size of the attractive basin of a locally exponentially stable equilibrium point is estimated. Finally, the utility of our results is illustrated via a numerical example.

Keywords: Cohen-Grossberg neural networks, Equilibrium point, Stability, Attractive basin.

1 Introduction

Recently, Cohen-Grossberg neural networks (CGNNs) have been an active research topic (see [1-6]), since it could include a lot of models from evolutionary, population biology and neurobiology. Especially, in the context of associative memory, a CGNNs should possess multiple locally asymptotically stable equilibrium points, and every equilibrium point that stands for a stored pattern should possess a guaranteed attractive basin. However, to our knowledge, few results have now been reported in the literatures on the local stability condition of an equilibrium point of a CGNNs, or the estimation of size of the attractive basin of a locally asymptotically stable equilibrium point of a CGNNs (see [1-4, 6]).

Motivated by the previous work and comment, this paper addresses the local stability of delayed CGNNs. A sufficient condition for the local exponential stability of an equilibrium point is presented, and an estimate on the size of the attractive basin of a locally exponentially stable equilibrium point is established. The proposed condition and estimate are easily checkable and applicable, because they are phrased in terms of the network parameters, the neurons' nonlinearities, and the relevant equilibrium point. To our knowledge, this is the first time to present such an estimate for delayed CGNNs. The utility of our results is illustrated via a numerical example.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 651–658, 2010.

2 Preliminaries

In this paper, we will investigate delayed Cohen-Grossberg neural networks of the form

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = a_i(x_i)\left[b_i(x_i) - \sum_{j=1}^{n} c_{ij}d_j(x_j) - \sum_{j=1}^{n} e_{ij}h_j(x_j(t-\tau_j(t)))\right], \, t \geq 0, \quad (2.1)$$

$i = 1, 2, \cdots, n$, where $x_i(t)$ stands for the state of the ith neuron at time t, $d_i(\cdot)$ and $h_i(\cdot)$ stand respectively for the output of the ith neuron at time t and $t - \tau_i(t)$, c_{ij} and e_{ij} are the connection weight from the jth neuron to the ith neuron.

In this paper, we adopt the following hypotheses.

(H_1) The function a_i is positive and continuous on R, $i = 1, 2, \cdots, n$.

(H_2) The function b_i is non-increasing and continuously differentiable on R, $i = 1, 2, \cdots, n$.

(H_3) The functions d_i and h_i are continuously differentiable on R, $i=1, 2, \cdots, n$.

(H_4) The time delays $\tau_i(t)$ are continuous and bounded with $0 \leq \tau_i(t) \leq \tau$, $i = 1, 2, \cdots, n$.

An equilibrium point of system (2.1) is a vector $x^* = (x_1^*, \cdots, x_n^*)^T$ satisfying

$$b_i(x_i^*) = \sum_{j=1}^{n} c_{ij}d_j(x_j^*) + \sum_{j=1}^{n} e_{ij}h_j(x_j^*), \quad i = 1, 2, \cdots, n.$$

Definition 2.1. Let x^* be an equilibrium point of system (2.1),
(a) x^* is locally stable if for every $\varepsilon > 0$, there exists $\delta > 0$, such that every solution x to (2.1) with $\|x(0) - x^*\|_\infty < \delta$ satisfies $\|x(t) - x^*\|_\infty < \varepsilon$ for all $t \geq 0$.
(b) x^* is locally exponentially stable if (i) x^* is locally stable; and (ii) there are three positive numbers δ, α, β, such that every solution x to with $\|x(0) - x^*\|_\infty < \delta$ satisfies $\|x(t) - x^*\|_\infty < \alpha \|x(0) - x^*\|_\infty e^{-\beta t}$ for all $t \geq 0$.
(c) The attractive basin of x^*, $AB(x^*)$, is a maximal region such that any solution x to (2.1) with $x(0) \in AB(x^*)$ approaches x^*.

The following properties of the upper right Dini derivative can readily be proved.

Lemma 2.1. Let f be a continuous function on R, g a differentiable function on R. Then

$$D^+(f(t) + g(t)) = D^+ f(t) + \dot{g}(t).$$

Lemma 2.2. Let f be a differentiable function on R. Then

$$D^+ |f(t)| = \begin{cases} \dot{f}(t), & \text{if } f(t) > 0; \\ -\dot{f}(t), & \text{if } f(t) < 0; \\ \left|\dot{f}(t)\right|, & \text{if } f(t) = 0. \end{cases}$$

Lemma 2.3. Let f and g be two continuous functions on R. Then

$$D^+ \max\{f(t), g(t)\} = \begin{cases} D^+ f(t), & \text{if } f(t) > g(t); \\ D^+ g(t), & \text{if } f(t) < g(t); \\ \max\{D^+ f(t), D^+ g(t)\}, & \text{if } f(t) = g(t). \end{cases}$$

Lemma 2.4. Let f be a differentiable function on R, and g be a continuous function on R. Then

$$D^+(f(t) \times g(t)) = \dot{f}(t)g(t) + f(t)D^+ g(t).$$

3 Main Results

Theorem 3.1. Consider a CGNNs of the form (2.1) that satisfies $(H_1) - (H_4)$. Let $x^* = (x_1^*, \cdots, x_n^*)^T$ be an equilibrium point of this network, such that

$$\left|\dot{b}_i(x_i^*)\right| > \sum_{j=1}^{n} |c_{ij}| \left|\dot{d}_j(x_j^*)\right| + \sum_{j=1}^{n} |e_{ij}| \left|\dot{h}_j(x_j^*)\right|, \quad i = 1, 2, \cdots, n. \tag{3.1}$$

Then x^* is locally exponentially stable, and

$$\{x \in R^n : \|x - x^*\|_\infty < \delta\} \subseteq AB(x^*), \tag{3.2}$$

where

$$\begin{aligned} \delta = \sup\{r : & \min_{|z - x_i^*| \le r} \{\left|\dot{b}_i(z)\right|\} > \sum_{j=1}^{n} |c_{ij}| \max_{|z - x_j^*| \le r} \{\left|\dot{d}_j(z)\right|\} \\ & + \sum_{j=1}^{n} |e_{ij}| \max_{|z - x_j^*| \le r} \{\left|\dot{h}_j(z)\right|\}\}, \quad i = 1, 2, \cdots, n. \end{aligned} \tag{3.3}$$

In order to prove this theorem, we need the following lemma.

Lemma 3.2. Consider a CGNNs of the form (2.1) that satisfies $(H_1) - (H_4)$. Let $x^* = (x_1^*, \cdots, x_n^*)^T$ be an equilibrium point of this network that satisfies the condition (3.1). Let $0 < r < \delta$, where δ is defined in (3.3). Let $\alpha > 0$ satisfy

$$\begin{aligned} \alpha < \min_{1 \le i \le n} \{ & \min_{|z - x_i^*| \le r} \{a_i(z)\} \Big[\min_{|z - x_i^*| \le r} \{\left|\dot{b}_i(z)\right|\} - \sum_{j=1}^{n} |c_{ij}| \max_{|z - x_j^*| \le r} \{\left|\dot{d}_j(z)\right|\} \\ & - \sum_{j=1}^{n} |e_{ij}| \max_{|z - x_j^*| \le r} \{\left|\dot{h}_j(z)\right|\} \Big]\}, \quad i = 1, 2, \cdots, n. \end{aligned} \tag{3.4}$$

Define a Lyapunov function associated with every solution $x \neq x^*$ of this network in the following way:

$$V(x(t)) = e^{\alpha t} \|x(t) - x^*\|_\infty = e^{\alpha t} \max_{1 \le i \le n} \{|x_i(t) - x_i^*|\}, \tag{3.5}$$

Then the following assertions hold.

(a) Given any $t \geq 0$. If $\|x(t) - x^*\|_\infty < r$, then $D^+V(x(t)) < 0$.
(b) Given any $t \geq 0$. If $\|x(0) - x^*\|_\infty < r$ and $V(x(t)) < V(x(0))$, then $\|x(t) - x^*\|_\infty < r$.
(c) The function $D^+V(x(t))$ is continuous.
(d) If $\|x(0) - x^*\|_\infty < r$, then $D^+V(x(t)) < 0$ holds for all $t \geq 0$.

Proof. (a) For any given $t \geq 0$, there is an integer k, $1 \leq k \leq n$, such that

$$|x_k - x_k^*| = \max_{1 \leq i \leq n} \{|x_i - x_i^*|, |x_i(t - \tau_i(t)) - x_i^*|\} > 0 \tag{3.6}$$

Clearly,

$$\begin{aligned}
\frac{\mathrm{d}(x_k - x_k^*)}{\mathrm{d}t} &= a_k(x_k)\Big[b_k(x_k) - \sum_{j=1}^{n} c_{kj} d_j(x_j) - \sum_{j=1}^{n} e_{kj} h_j(x_j(t - \tau_j(t)))\Big] \\
&\quad - a_k(x_k)\Big[b_k(x_k^*) - \sum_{j=1}^{n} c_{kj} d_j(x_j^*) - \sum_{j=1}^{n} e_{kj} h_j(x_j^*)\Big] \\
&= a_k(x_k)\big[b_k(x_k) - b_k(x_k^*)\big] - a_k(x_k) \sum_{j=1}^{n} c_{kj}\big[d_j(x_j) - d_j(x_j^*)\big] \\
&\quad - a_k(x_k) \sum_{j=1}^{n} e_{kj}\big[h_j(x_j(t - \tau_j(t))) - h_j(x_j^*)\big]
\end{aligned} \tag{3.7}$$

By Lemma 2.2 and in view of (H_2), we get

$$\begin{aligned}
D^+ |x_k - x_k^*| &\leq -a_k(x_k) |b_k(x_k) - b_k(x_k^*)| + a_k(x_k) \sum_{j=1}^{n} |c_{kj}| \left|d_j(x_j) - d_j(x_j^*)\right| \\
&\quad + a_k(x_k) \sum_{j=1}^{n} |e_{kj}| \left|h_j(x_j(t - \tau_j(t))) - h_j(x_j^*)\right|
\end{aligned} \tag{3.8}$$

By the Lagrange mean value theorem, we get

$$|b_k(x_k) - b_k(x_k^*)| = \left|\dot{b}_k(\xi_k)\right| |x_k - x_k^*|, \tag{3.9}$$

where ξ_k lies between x_k^* and x_k.

By the Lagrange mean value theorem and in view of (3.6), we get

$$\left|d_j(x_j) - d_j(x_j^*)\right| = \left|\dot{d}_j(\zeta_j)\right| |x_j - x_j^*| \leq \left|\dot{d}_j(\zeta_j)\right| |x_k - x_k^*|, \quad j = 1, 2, \cdots, n. \tag{3.10}$$

where ζ_j lies between x_j^* and x_j.

$$\left|h_j(x_j(t - \tau_j(t))) - h_j(x_j^*)\right| = \left|\dot{h}_j(\eta_j)\right| |x_j(t - \tau_j(t)) - x_j^*| \leq \left|\dot{h}_j(\eta_j)\right| |x_k - x_k^*|, \quad j = 1, 2, \cdots, n. \tag{3.11}$$

where η_j lies between x_j^* and x_j.

Substituting (3.9) – (3.11) into (3.8), we get

$$D^+ |x_k - x_k^*| \le -a_k(x_k)\Big[\Big|\dot{b}_k(\xi_k)\Big| - \sum_{j=1}^{n} |c_{kj}| \Big|\dot{d}_j(\zeta_j)\Big| - \sum_{j=1}^{n} |e_{kj}| \Big|\dot{h}_j(\eta_j)\Big| \Big] |x_k - x_k^*| \tag{3.12}$$

By virtue of (3.6) and (3.12), it follows from Lemma 2.4 that

$$\begin{aligned}
D^+V(x) &= \alpha e^{\alpha t} \max_{1\le i\le n} \{|x_i - x_i^*|\} + e^{\alpha t} D^+\big(\max_{1\le i\le n} \{|x_i - x_i^*|\} \big) \\
&= \alpha e^{\alpha t} |x_k - x_k^*| + e^{\alpha t} D^+ |x_k - x_k^*| \\
&\le -e^{\alpha t}\{a_k(x_k)\big[\Big|\dot{b}_k(\xi_k)\Big| - \sum_{j=1}^{n} |c_{kj}| \Big|\dot{d}_j(\zeta_j)\Big| - \sum_{j=1}^{n} |e_{kj}| \Big|\dot{h}_j(\eta_j)\Big| \big] \\
&\quad - \alpha\} |x_k - x_k^*| \\
&\le -e^{\alpha t}\{ \min_{|z-x_k^*|\le r} \{a_k(z)\} \Big[\min_{|z-x_k^*|\le r} \{\Big|\dot{b}_k(z)\Big|\} - \sum_{j=1}^{n} |c_{kj}| \max_{|z-x_j^*|\le r} \{\Big|\dot{d}_j(z)\Big|\} \\
&\quad - \sum_{j=1}^{n} |e_{kj}| \max_{|z-x_j^*|\le r} \{\Big|\dot{h}_j(z)\Big|\} \Big] - \alpha\} |x_k - x_k^*| \\
&< 0.
\end{aligned}$$

(b) Clearly,

$$\|x(t) - x^*\|_\infty = e^{-\alpha t} V(x(t)) \le V(x(t)) \le V(x(0)) = \|x(0) - x^*\|_\infty < r.$$

(c) By sequentially applying Lemma 2.2, Lemma 2.3 and Lemma 2.4, we get

$$\begin{aligned}
D^+V(x(t)) &= \alpha e^{\alpha t} \max_{1\le i\le n} \{|x_i(t) - x_i^*|\} + e^{\alpha t} \max_{1\le k\le n} \left\{D^+ |x_k(t) - x_k^*|\right\} \\
&= \alpha e^{\alpha t} \max_{1\le i\le n} \{|x_i(t) - x_i^*|\} \\
&\quad + e^{\alpha t} \max\left\{ \max_{1\le k\le n, x_k(t) > x_k^*} \{\dot{x}_k(t)\}, \max_{1\le k\le n, x_k(t) < x_k^*} \{-\dot{x}_k(t)\} \right\}.
\end{aligned}$$

It follows from the continuity of x_i and $\dfrac{\mathrm{d}x_i}{\mathrm{d}t}$ (see (2.1)), that the function $D^+V(x(t))$ is continuous.

(d) From $\|x(0) - x^*\|_\infty < r$ and by Lemma 3.2(a), we induce $D^+V(x(0)) < 0$. For the sake of contradiction, we assume there is $t > 0$ such that $D^+V(x(t)) \ge 0$. Let $t_0 = \inf\{t : D^+V(x(t)) \ge 0\}$. It follows from Lemma 3.2(c), that

$$D^+V(x(t_0)) \ge 0. \tag{3.13}$$

Thus $D^+V(x(t)) < 0$ for all $t \in [0, t_0]$, which implies $V(x(t))$ is strictly decreasing in the interval $[0, t_0]$ and hence $V(x(t_0)) < V(x(0))$. From it, and by Lemma 3.2(b), we get $\|x(t_0) - x^*\|_\infty < r$. This inequality plus Lemma 3.2(a) yields

$$D^+V(x(t_0)) < 0. \tag{3.14}$$

There is a contradiction between the (3.13) and (3.14).

We are now in a position to complete the proof of Theorem 3.1.

Proof of Theorem 3.1. Let $x \neq x^*$ be an arbitrary solution to the network, such that $\|x(0) - x^*\|_\infty < \delta$. Then there exists $r > 0$ such that $\|x(0) - x^*\|_\infty < r < \delta$. Hence,

$$\min_{1\le i\le n}\Big\{\min_{|z-x_i^*|\le r}\{a_i(z)\}\Big[\min_{|z-x_i^*|\le r}\{\left|\dot b_i(z)\right|\} - \sum_{j=1}^{n}|c_{ij}|\max_{|z-x_j^*|\le r}\{\left|\dot d_j(z)\right|\} - \sum_{j=1}^{n}|e_{ij}|\max_{|z-x_j^*|\le r}\{\left|\dot h_j(z)\right|\}\Big]\Big\} > 0, \quad i = 1, 2, \cdots, n.$$

Let $\alpha > 0$ satisfy

$$\alpha < \min_{1\le i\le n}\Big\{\min_{|z-x_i^*|\le r}\{a_i(z)\}\Big[\min_{|z-x_i^*|\le r}\{\left|\dot b_i(z)\right|\} - \sum_{j=1}^{n}|c_{ij}|\max_{|z-x_j^*|\le r}\{\left|\dot d_j(z)\right|\} - \sum_{j=1}^{n}|e_{ij}|\max_{|z-x_j^*|\le r}\{\left|\dot h_j(z)\right|\}\Big]\Big\}, \quad i = 1, 2, \cdots, n.$$

and let

$$V(x(t)) = e^{\alpha t}\|x(t) - x^*\|_\infty = e^{\alpha t}\max_{1\le i\le n}\{|x_i(t) - x_i^*|\}.$$

It follows from Lemma 3.2(d) that $V(x(t)) < V(x(0))$ for all $t \ge 0$. So,

$$\|x(t) - x^*\|_\infty = e^{-\alpha t}V(x(t)) \le e^{-\alpha t}V(x(0)) = e^{-\alpha t}\|x(0) - x^*\|_\infty.$$

The claimed assertions follow.

As a byproduct of Theorem 3.1, we have

Theorem 3.3. Consider a CGNNs of the form (2.1) that satisfies $(H_1) - (H_4)$. Let $x^* = (x_1^*, \cdots, x_n^*)^T$ be an equilibrium point of this network that satisfies

$$\inf_{z\in R}\left\{\left|\dot b_i(z)\right|\right\} > \sum_{j=1}^{n}|c_{ij}|\sup_{z\in R}\left\{\left|\dot d_j(z)\right|\right\} + \sum_{j=1}^{n}|e_{ij}|\sup_{z\in R}\left\{\left|\dot h_j(z)\right|\right\}, \ i = 1, 2, \cdots, n. \tag{3.15}$$

Then x^* is globally exponentially stable.

4 Application

As the application to neural associative memories, we consider the following example.

Example. Consider the CGNNs

$$\begin{cases} \dfrac{\mathrm{d}x_1}{\mathrm{d}t} = b_1(x_1) - c_{11}d_1(x_1), \\ \dfrac{\mathrm{d}x_2}{\mathrm{d}t} = b_2(x_2) - c_{21}d_1(x_1) - c_{22}d_2(x_2) - c_{22}d_1(x_1(t-\tau(t))), \\ \dfrac{\mathrm{d}x_3}{\mathrm{d}t} = b_3(x_3) - c_{31}d_1(x_1) - c_{33}d_3(x_3) - e_{33}d_1(x_1(t-\tau(t))), \\ \dfrac{\mathrm{d}x_4}{\mathrm{d}t} = b_4(x_4) - c_{41}d_1(x_1) - c_{44}d_4(x_4), \end{cases} \tag{4.1}$$

where $d_i(z) = d(z) = 2\tanh(z), 1 \le i \le 4$, $\tau(t) = \sin^2(t)$, $\alpha = d(1) = -d(-1)$,

$$b_1(x_1) = e_1x_1^3 - e_1 + \alpha c_{11}, \qquad b_2(x_2) = \alpha c_{22}x_2^3 + \alpha(c_{21} + c_{22}),$$
$$b_3(x_3) = \alpha c_{33}x_3^3 + \alpha(c_{31} + e_{33}), \quad b_4(x_4) = \alpha c_{44}x_4^3 + \alpha c_{41}.$$

By simple calculation, we get $\dot{d}(z) = 2\,\mathrm{sech}^2(z)$, $\beta = \left|\dot{d}(1)\right| = \left|\dot{d}(-1)\right| = 0.8399$.

It can be verified that the four patterns

$$x^{(1)} = (1,1,1,1)^T, \qquad x^{(2)} = (1,1,-1,-1)^T,$$
$$x^{(3)} = (1,-1,-1,1)^T, \quad x^{(4)} = (1,-1,1,-1)^T$$

are all equilibrium points of the network (4.1).

If the conditions

$$e_1, c_{22}, c_{33}, c_{44} < 0, \quad 3|e_1| > \beta|c_{11}|, \quad 3\alpha|c_{22}| > \beta|c_{21}| + 2\beta|c_{22}|,$$
$$3\alpha|c_{33}| > \beta|c_{31}| + \beta|c_{33}| + \beta|e_{33}|, \qquad 3\alpha|c_{44}| > \beta|c_{41}| + \beta|c_{44}|$$

hold simultaneously, it follows from Theorem 3.1 that these four equilibrium points are all locally exponentially stable. Furthermore,

$$\{x \in R^4 : \left\|x - x^{(i)}\right\|_\infty < \delta\} \subseteq AB(x^{(i)}), \quad 1 \le i \le 4,$$

where $\delta = \min\{\delta_1, \delta_2, \delta_3, \delta_4\}$, $0 < \delta_i < 1$, and

$$3|e_1|(1-\delta_1)^2 = 2|c_{11}|\,\mathrm{sech}^2(1+\delta_1),$$
$$3\alpha|c_{22}|(1-\delta_2)^2 = 2(|c_{21}|) + 2|c_{22}|)\mathrm{sech}^2(1+\delta_2),$$
$$3\alpha|c_{33}|(1-\delta_3)^2 = 2(|c_{31}|) + |c_{33}| + |e_{33}|)\mathrm{sech}^2(1+\delta_3),$$
$$3\alpha|c_{44}|(1-\delta_4)^2 = 2(|c_{41}|) + |c_{44}|)\mathrm{sech}^2(1+\delta_4).$$

As an instance, let us set $e_1 = -\alpha$, $c_{22} = c_{33} = c_{44} = -1$, $c_{11} = 1.5$, $c_{21} = -0.5$, $c_{31} = 0.25$, $e_{33} = 0.25$, $c_{41} = -0.5$. Then the system (4.1) reduces to

$$\begin{cases} \dfrac{\mathrm{d}x_1}{\mathrm{d}t} = -1.5232(x_1^3 - 2.5) - 1.5d(x_1), \\ \dfrac{\mathrm{d}x_2}{\mathrm{d}t} = -1.5232(x_2^3 + 0.5) + 0.5d(x_1) + d(x_2) + d(x_1(t - \sin^2(t))), \\ \dfrac{\mathrm{d}x_3}{\mathrm{d}t} = -1.5232(x_3^3 - 0.5) - 0.25d(x_1) + d(x_3) - 0.25d(x_1(t - \sin^2(t))), \\ \dfrac{\mathrm{d}x_4}{\mathrm{d}t} = -1.5232(x_4^3 + 0.5) + 0.5d(x_1) + d(x_4). \end{cases} \tag{4.2}$$

The previous argument ensures that $x^{(1)}, x^{(2)}, x^{(3)}$ and $x^{(4)}$ are all locally exponentially stable equilibrium points of this system. Furthermore,

$$\{x \in R^4 : \left\|x - x^{(i)}\right\|_\infty < 0.7184\} \subseteq AB(x^{(i)}), \quad 1 \le i \le 4.$$

Acknowledgements. The work is supported by Key Science Foundation of Educational Department of Hubei Province under Grant D20082201 and Z20062202, Innovation Teams of Hubei Normal University.

References

1. Li, C., Yang, S.: Existence and Attractivity of Periodic Solutions to Non-autonomous Cohen-Grossberg Neural Networks with Time Delays. Chaos, Solitons and Fractals 41, 1235–1244 (2009)
2. Tan, M., Zhang, Y.: New Sufficient Conditions for Global Asymptotic Stability of Cohen-Grossberg Neural Networks with Time-Varying Delays. Nonlinear Analysis: Real World Applications 10, 2139–2145 (2009)
3. Xiang, H., Cao, J.: Almost Periodic Solution of Cohen-Grossberg Neural Networks with Bounded and Unbounded Delays. Nonlinear Analysis: Real World Applications 10, 2407–2419 (2009)
4. Zhou, Q., Wan, L.: Impulsive Effects on Stability of Cohen-Grossberg-Type Bidirectional Associative Memory Neural Networks with Delays. Nonlinear Analysis: Real World Applications 10, 2531–2540 (2009)
5. Yang, X., Liao, X., Li, C., Tang, Y.: Local Stability and Attractive Basin of Cohen-Grossberg Neural Networks. Nonlinear Analysis: Real World Applications 10, 2834–2841 (2009)
6. Yang, X.: Existence and Global Exponential Stability of Periodic Solution for Cohen-Grossberg Shunting Inhibitory Cellular Neural Networks with Delays and Impulses. Neurocomputing 72, 2219–2226 (2009)

Exponential Stability Analysis for Discrete-Time Stochastic BAM Neural Networks with Time-Varying Delays

Tiheng Qin[1], Quanxiang Pan[2], and Yonggang Chen[2]

[1] The Basic Department, Henan Mechanical and Electrical Engineering College, Xinxiang 453002, China
[2] Department of Mathematics, Henan Institute of Science and Technology, Xinxiang 453003, China
tihengqin2008@tom.com, happycygzmd@tom.com

Abstract. In this paper, the global exponential stability analysis problem is investigated for discrete-time stochastic BAM neural networks with time-varying delays. By constructing the appropriate Lyapunov functional, and by resorting to free weight matrix method, the delay-dependent exponential stability criteria are derived in terms of linear matrix inequalities (LMIs). Numerical examples are presented to show the effectiveness and benefits of ours results.

Keywords: Exponential stability; Discrete-time BAM neural networks; Stochastic disturbances; Time-varying delays; LMIs.

1 Introduction

Due to the wide applications of bidirectional associative memory (BAM) neural networks and the existence of time delays in many practical neural networks, the stability analysis problem for continuous-time delayed BAM neural networks has received great attention during the past years, see. e.g. [1-3]. However, when implementing the continuous-time neural networks for computer simulation, for experimental or computational purposes, it is essential to formulate a discrete-time system that is an analogue of the continuous-time recurrent neural networks. Therefore, the stability analysis for discrete-time delayed BAM neural networks was also widely studied in recent years [4-7]. By employing the more general Lyapunov functional and by introducing some slack matrices, the improved delay-dependent exponential stability criterion was proposed in [7] in terms of LMIs. However, it should be pointed out that $h(k), h_M - h(k), \tau(k), \tau_M - \tau(k)$ in [7] were enlarged as $h_M, h_M - h_m, \tau_M, \tau_M - \tau_m$, respectively, which may bring much conservativeness. On the other hand, in real nervous systems, the synaptic transmission is a noisy process brought on by random fluctuations from the release of neurotransmitters and other probabilistic causes, and it has been realized that a neural network could be stabilized or destabilized by certain stochastic inputs [8] which leads to the research on dynamics of stochastic neural networks, see,

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 659–666, 2010.

e.g. [9-10]. However, to the best of the authors' knowledge, the exponential stability analysis problem for discrete-time stochastic BAM neural networks with time-varying delays has not been adequately investigated.

Motivated by the above discussions, this paper considers the exponential stability problem for a class of discrete-time stochastic BAM neural networks with time-varying delays. By constructing the appropriate Lyapunov functional, and by using the less conservative enlargement for estimating the difference of Lyapunov functional, we obtain the delay-dependent exponential stability criteria in terms of LMIs. Finally, two numerical examples are given to illustrate the effectiveness and less conservativeness of the obtained results.

Notation. Throughout this paper, the superscript "T" stands for the transpose of a matrix. $\mathbb{R}^n$ and $\mathbb{R}^{n\times n}$ denote the n-dimensional Euclidean space and set of all $n \times n$ real matrices, respectively. A real symmetric matrix $P > 0 (\geq 0)$ denotes P being a positive definite (positive semi-definite) matrix. I is used to denote an identity matrix with proper dimension. For integers a, b, and $a < b$, $N[a, b]$ denotes the discrete interval $\mathbb{N}[a, b] = \{a, a + 1, \cdots, b - 1, b\}$. $\lambda_M(X)$ and $\lambda_m(X)$ stand for the maximum and minimum eigenvalues of the symmetric matrix X, respectively. $\|\cdot\|$ refers to the Euclidean vector norm. $(\Omega, \mathcal{F}, \mathcal{P})$ denotes a complete probability with a filtration $\mathcal{F}_t \geq 0$ satisfying the usual conditions (i.e., the filtration contains all $\mathcal{P}$-null sets and is right continuous). $\mathbb{E}\{\cdot\}$ stands for the mathematical expectation operator with respect to the given probability measure $\mathcal{P}$. Matrices, if not explicitly stated, are assumed to have compatible dimensions. The symmetric terms in a symmetric matrix are denoted by $*$.

2 Problem Formulation

In this paper, the discrete-time stochastic delayed BAM neural network considered can be described as follows:

$$\begin{aligned}
&x(k+1) = Ax(k) + Wf(y(k-\tau(k))) + \sigma_1(k, x(k), y(k-\tau(k)))\omega(k),\\
&y(k+1) = By(k) + Vg(x(k-h(k))) + \sigma_2(k, y(k), x(k-h(k)))\omega(k),\\
&x(s) = \phi(s),\ y(s) = \psi(s),\ \forall\ s \in \mathbb{N}[-\tau^*, 0],\ \tau^* = max\{\tau_M, h_M\},
\end{aligned} \tag{1}$$

where $x(k) = [x_1(k), x_2(k), \cdots, x_n(k)]^T$ and $y(k) = [y_1(k), y_2(k), \cdots, y_m(k)]^T$ denote the states of the neurons; $A = \text{diag}\{a_1, a_2, \cdots, a_n\}$ and $B = \text{diag}\{b_1, b_2, \cdots, b_m\}$ with $a_i, b_j \in (0,\ 1)$; $W = (w_{ij})_{m\times n}$, $V = (v_{ij})_{n\times m}$ and w_{ij}, v_{ji} denote the synaptic connection weights; $f(\cdot) = [f_1(\cdot), f_2(\cdot), \cdots, f_m(\cdot)]^T$, $g(\cdot) = [g_1(\cdot), g_2(\cdot), \cdots, g_n(\cdot)]^T$. and $f_j(\cdot), g_i(\cdot)$ denote the activation functions of the neurons; $\omega(k)$ is a scalar Brownian motion defined on $(\Omega, \mathcal{F}, \mathcal{P})$ with $\mathbb{E}\{\omega(k)\} = 0, \mathbb{E}\{\omega^T(k)\omega(k)\} = 1$ and $\mathbb{E}\{\omega^T(i)\omega(j)\} = 0$ $(i \neq j)$; the functions $\phi(s), \psi(s)$ represent initial conditions; $\tau(k)$ and $h(k)$ represent time-varying delays satisfying $\tau_m \leq \tau(k) \leq \tau_M$, $h_m \leq h(k) \leq h_M$, where τ_m, τ_M, h_m and h_M are positive integers. $\sigma_1 : \mathbb{R}\times\mathbb{R}^n\times\mathbb{R}^m \longrightarrow \mathbb{R}^n$, $\sigma_2 : \mathbb{R}\times\mathbb{R}^m\times\mathbb{R}^n \longrightarrow \mathbb{R}^m$ are the continuous functions, and are assumed to satisfy

$$\sigma_1^T\sigma_1 \leq \rho_1 x^T(k)x(k) + \rho_2 y^T(k-\tau(k)))y(k-\tau(k))), \tag{2}$$

$$\sigma_2^T\sigma_2 \le \rho_3 y^T(k)y(k) + \rho_4 x^T(k-h(k)))x(k-h(k))), \tag{3}$$

where $\rho_i (i = 1, 2, 3, 4)$ are known constant scalars. Throughout this paper, the the activation functions $f_j(\cdot),\ g_i(\cdot)$ satisfy the following condition
(A1) For any $\zeta \in R$, there exist positive scalars $l_{1j},\ l_{2i}$ such that

$$0 \le |\ f_j(\zeta)\ | \le l_{1j}\ |\ \zeta\ |,\ 0 \le |\ g_i(\zeta)\ | \le l_{2i}\ |\ \zeta\ |\ . \tag{4}$$

Definition 1. The trivial solution of discrete-time stochastic BAM neural network (1) is said to be globally exponentially stable in the mean square, if there exist scalars $r > 1$ and $K > 1$ such that

$$\begin{aligned}&\mathbb{E}\{\|x(k)\|^2\} + \mathbb{E}\{\|y(k)\|^2\}\\ &\le K\big(\sup_{s\in\mathbb{N}[-h_M,0]}\mathbb{E}\{\|\phi(s)\|^2\} + \sup_{s\in\mathbb{N}[-\tau_M,0]}\mathbb{E}\{\|\psi(s)\|^2\}\big)r^{-k}.\end{aligned}$$

3 Main Results

Theorem 1. For given matrices $L_1 = \mathrm{diag}\{l_{11}, \cdots, l_{1m}\}, L_2 = \mathrm{diag}\{l_{21}, \cdots, l_{2n}\}$, and constants h_m, h_M, τ_m, τ_M. The discrete-time stochastic delayed BAM neural network (6) is globally exponentially stable in the mean square, if there exist positive scalars λ_1 and λ_2, matrices $P_i > 0, Q_i > 0, R_i > 0, S_i > 0, Z_i > 0, (i = 1, 2), L_j, M_j, N_j, T_j, (j = 1, 2, \cdots, 8)$, and diagonal matrices $D_1 = \mathrm{diag}\{d_{11}, \cdots, d_{1m}\} > 0, D_2 = \mathrm{diag}\{d_{21}, \cdots, d_{2n}\} > 0$, such that the following LMIs hold

$$rP_1 + r^{h_M}h_M Z_1 \le \lambda_1 I,\ rP_2 + r^{\tau_M}\tau_M Z_2 \le \lambda_2 I, \tag{5}$$

$$\begin{bmatrix}\Omega & \Sigma_1\\ \Sigma_1^T & \Sigma_2\end{bmatrix} < 0,\ \begin{bmatrix}\Omega & \Sigma_3\\ \Sigma_3^T & \Sigma_4\end{bmatrix} < 0,\ \begin{bmatrix}\Omega & \Sigma_5\\ \Sigma_5^T & \Sigma_6\end{bmatrix} < 0,\ \begin{bmatrix}\Omega & \Sigma_7\\ \Sigma_7^T & \Sigma_8\end{bmatrix} < 0, \tag{6}$$

where

$$\Omega = \begin{bmatrix}\Omega_{11} & \Omega_{12} & \Omega_{13} & \Omega_{14} & \Omega_{15} & \Omega_{16} & \Omega_{17} & L_8^T\\ * & \Omega_{22} & \Omega_{23} & \Omega_{24} & \Omega_{25} & \Omega_{26} & \Omega_{27} & \Omega_{28}\\ * & * & \Omega_{33} & -T_4^T & \Omega_{35} & \Omega_{36} & \Omega_{37} & -T_8^T\\ * & * & * & \Omega_{44} & N_4 & \Omega_{46} & -M_4 & 0\\ * & * & * & * & \Omega_{55} & \Omega_{56} & \Omega_{57} & \Omega_{58}\\ * & * & * & * & * & \Omega_{66} & \Omega_{67} & \Omega_{68}\\ * & * & * & * & * & * & \Omega_{77} & -M_8^T\\ * & * & * & * & * & * & * & \Omega_{88}\end{bmatrix},$$

$$\begin{aligned}&\Sigma_1 = \left[\sqrt{h_M}L\ \sqrt{\tau_M}N\right], \Sigma_3 = \left[\sqrt{h_M}L\ \sqrt{\tau_m}N\ \sqrt{\tau_M-\tau_m}T\right],\\ &\Sigma_2 = \mathrm{diag}\{-Z_1, -Z_2\}, \Sigma_4 = \mathrm{diag}\{-Z_1, -Z_2, -Z_2\},\\ &\Sigma_5 = \left[\sqrt{h_m}L\ \sqrt{h_M-h_m}M\ \sqrt{\tau_M}N\right], \Sigma_6 = \mathrm{diag}\{-Z_1, -Z_1, -Z_2\},\\ &\Sigma_7 = \left[\sqrt{h_m}L\ \sqrt{h_M-h_m}M\ \sqrt{\tau_m}N\ \sqrt{\tau_M-\tau_m}T\right],\\ &\Sigma_8 = \mathrm{diag}\{-Z_1, -Z_1, -Z_2, -Z_2\},\end{aligned}$$

and

$\Omega_{11} = rA^TP_1A + r^{h_M}h_M(A-I)^TZ_1(A-I) - P_1 + (h_M - h_m + 1)Q_1 + R_1$
$+ L_1 + L_1^T + \lambda_1\rho_1 I,\ \Omega_{12} = L_2^T + T_1 - N_1,\ \Omega_{13} = L_3^T - T_1,\ \Omega_{14} = rA^TP_1W$
$+ r^{h_M}h_M(A-I)^TZ_1W + L_4^T,\ \Omega_{15} = L_5^T + N_1,\ \Omega_{16} = -L_1 + L_6^T + M_1,$
$\Omega_{17} = -M_1 + L_7^T,\ \Omega_{22} = -r^{-\tau_M}Q_2 + T_2 + T_2^T + D_1 - N_2 - N_2^T + \lambda_1\rho_2 I,$
$\Omega_{23} = -T_2 + T_3^T - N_3^T,\ \Omega_{24} = T_4^T - N_4^T,\ \Omega_{25} = T_5^T + N_2 - N_5^T,\ \Omega_{26} = T_6^T$
$+ M_2 - L_2 - N_6^T,\ \Omega_{27} = -M_2 + T_7^T - N_7^T,\ \Omega_{28} = T_8^T - N_8^T,\ \Omega_{33} = -r^{-\tau_M}R_2$
$- T_3 - T_3^T,\ \Omega_{35} = N_3 - T_5^T,\ \Omega_{36} = M_3 - L_3 - T_6^T,\ \Omega_{37} = -M_3 - T_7^T,\ \Omega_{44} =$
$rW^TP_1W + r^{h_M}h_MW^TZ_1W - L_1^{-1}D_1L_1^{-1},\ \Omega_{46} = M_4 - L_4,\ \Omega_{55} = rB^TP_2B +$
$r^{\tau_M}\tau_M(B-I)^TZ_2(B-I) - P_2 + (\tau_M - \tau_m + 1)Q_2 + R_2 + N_5 + N_5^T + \lambda_2\rho_3 I,$
$\Omega_{56} = M_5 - L_5 + N_6^T,\ \Omega_{57} = -M_5 + N_7^T,\ \Omega_{58} = rB^TP_2V + r^{\tau_M}\tau_M(B-I)^T$
$\times Z_2V + N_8^T,\ \Omega_{66} = -r^{-h_M}Q_1 - L_6 - L_6^T + D_2 + M_6 + M_6^T + \lambda_2\rho_4 I,$
$\Omega_{67} = -L_7^T - M_6 + M_7^T,\ \Omega_{68} = -L_8^T + M_8^T,\ \Omega_{77} = -r^{-h_M}R_1 - M_7 - M_7^T,$
$\Omega_{88} = rV^TP_2V + r^{\tau_M}\tau_MV^TZ_2V - L_2^{-1}D_2L_2^{-1},\ L = \begin{bmatrix} L_1^T & L_2^T & \dots & L_8^T \end{bmatrix}^T,$
$M = \begin{bmatrix} M_1^T & M_2^T \cdots & M_8^T \end{bmatrix}^T,\ N = \begin{bmatrix} N_1^T & N_2^T & \cdots & N_8^T \end{bmatrix}^T,\ T = \begin{bmatrix} T_1^T & T_2^T & \cdots & T_8^T \end{bmatrix}^T.$

Proof. Choose the following Lyapunov-Krasovskii candidate function:

$$V(k) = V_1(k) + V_2(k) + V_3(k) + V_4(k) + V_5(k), \tag{7}$$

where

$$V_1(k) = r^k x^T(k)P_1x(k) + r^k y^T(k)P_2y(k),$$
$$V_2(k) = \sum_{l=k-h(k)}^{k-1} r^l x^T(l)Q_1x(l) + \sum_{l=k-\tau(k)}^{k-1} r^l y^T(l)Q_2y(l),$$
$$V_3(k) = \sum_{\theta=-h_M+1}^{-h_m}\sum_{l=k+\theta}^{k-1} r^l x^T(l)Q_1x(l) + \sum_{\theta=-\tau_M+1}^{-\tau_m}\sum_{l=k+\theta}^{k-1} r^l y^T(l)Q_2y(l),$$
$$V_4(k) = \sum_{l=k-h_M}^{k-1} r^l x^T(l)R_1x(l) + \sum_{l=k-\tau_M}^{k-1} r^l y^T(l)R_2y(l),$$
$$V_5(k) = r^{h_M}\sum_{\theta=-h_M}^{-1}\sum_{l=k+\theta}^{k-1} r^l\eta_1^T(l)Z_1\eta_1(l) + r^{\tau_M}\sum_{\theta=-\tau_M}^{-1}\sum_{l=k+\theta}^{k-1} r^l\eta_2^T(l)Z_2\eta_2(l),$$

and $\eta_1(l) = x(l+1) - x(l),\ \eta_2(l) = y(l+1) - y(l)$. Calculating the difference of $V(k)$ along the trajectories of (1), and taking the mathematical expectation, then we can obtain

$$\mathbb{E}\{\triangle V_1(k)\} = r^k\mathbb{E}\big\{x^T(k)(rA^TP_1A - P_1)x(k) + 2rx^T(k)A^TP_1Wf(y(k-\tau(k)))$$
$$+ rf^T(y(k-\tau(k))W^TP_1Wf(y(k-\tau(k)) + y^T(k)(rB^TP_2B$$

$$
\begin{aligned}
&- P_2)y(k) + 2ry^T(k)B^TP_2Vg(x(k-h(k)) + rg^T(x(k-h(k))\\
&\times V^TP_2Vg(x(k-h(k)) + r\sigma_1^TP_1\sigma_1 + r\sigma_2^TP_2\sigma_2\big\}, \qquad (8)\\
\mathbb{E}\{\triangle V_2(k)\}+&\mathbb{E}\{\triangle V_3(k)\} \le r^k\mathbb{E}\big\{(h_M - h_m + 1)x^T(k)Q_1x(k)\\
&- r^{-h_M}x^T(k-h(k))Q_1x(k-h(k)) + (\tau_M - \tau_m + 1)\\
&\times y^T(k)Q_2y(k) - r^{-\tau_M}y^T(k-\tau(k))Q_2y(k-\tau(k))\big\}, \qquad (9)\\
\mathbb{E}\{\triangle V_4(k)\} =&r^k\mathbb{E}\big\{x^T(k)R_1x(k) - r^{-h_M}x^T(k-h_M)R_1x(k-h_M)\\
&+ y^T(k)R_2y(k) - r^{-\tau_M}y^T(k-\tau_M)R_2y(k-\tau_M)\big\}, \qquad (10)\\
\mathbb{E}\{\triangle V_5(k)\} \le&r^k\mathbb{E}\big\{r^{h_M}h_M[(A-I)x(k) + Wf(y(k-\tau(k)))]^TZ_1[(A-I)x(k)\\
&+ Wf(y(k-\tau(k)))] + r^{\tau_M}\tau_M[(B-I)y(k) + Vg(x(k-h(k)))]^T\\
&\times Z_2[(B-I)y(k) + Vg(x(k-h(k)))]\\
&+ r^{h_M}h_M\sigma_1^T(k,x(k),y(k-\tau(k)))Z_1\sigma_1(k,x(k),y(k-\tau(k)))\\
&+ r^{\tau_M}\tau_M\sigma_2^T(k,y(k),x(k-h(k)))P_2\sigma_2(k,y(k),x(k-h(k)))\\
&- \sum_{l=k-h_M}^{k-1}\eta_1^T(l)Z_1\eta_1(l) - \sum_{l=k-\tau_M}^{k-1}\eta_2^T(l)Z_2\eta_2(l)\big\}. \qquad (11)
\end{aligned}
$$

Using the fact $2a^Tb \le a^TQa + b^TQ^{-1}b, (Q>0)$, we have

$$
\begin{aligned}
&- \sum_{l=k-h(k)}^{k-1}\eta_1^T(l)Z_1\eta_1(l) \le 2\sum_{l=k-h(k)}^{k-1}\eta_1^T(l)L^T\xi(k) + \sum_{l=k-h(k)}^{k-1}\xi^T(k)LZ_1^{-1}L^T\xi(k)\\
&= \xi^T(k)(\Psi_1L^T + L\Psi_1)\xi(k) + h(k)\xi^T(k)LZ_1^{-1}L^T\xi(k), \qquad (12)\\
&- \sum_{l=k-h_M}^{k-h(k)}\eta_1^T(l)Z_1\eta_1(l) \le 2\sum_{l=k-h_M}^{k-h(k)}\eta_1^T(l)M^T\xi(k) + \sum_{l=k-h_M}^{k-h(k)}\xi^T(k)MZ_1^{-1}M^T\xi(k)\\
&= \xi^T(k)(\Psi_2M^T + M\Psi_2)\xi(k) + (h_M - h(k))\xi^T(k)MZ_1^{-1}M^T\xi(k), \qquad (13)\\
&- \sum_{l=k-\tau(k)}^{k-1}\eta_2^T(l)Z_2\eta_2(l) \le 2\sum_{l=k-\tau(k)}^{k-1}\eta_2^T(l)N^T\xi(k) + \sum_{l=k-\tau(k)}^{k-1}\xi^T(k)NZ_2^{-1}N^T\xi(k)\\
&= \xi^T(k)(\Psi_3N^T + N\Psi_3)\xi(k) + \tau(k)\xi^T(k)NZ_2^{-1}N^T\xi(k), \qquad (14)\\
&- \sum_{l=k-\tau_M}^{k-\tau(k)}\eta_2^T(l)Z_2\eta_2(l) \le 2\sum_{l=k-\tau_M}^{k-\tau(k)}\eta_2^T(l)T^T\xi(k) + \sum_{l=k-\tau_M}^{k-\tau(k)}\xi^T(k)TZ_2^{-1}T^T\xi(k)\\
&= \xi^T(k)(\Psi_4T^T + T\Psi_4)\xi(k) + (\tau_M - \tau(k))\xi^T(k)TZ_2^{-1}T^T\xi(k), \qquad (15)
\end{aligned}
$$

where L, M, N, T are defined in Theorem 1 and

$$
\begin{aligned}
\Psi_1 &= \begin{bmatrix} I & 0 & 0 & 0 & 0 & -I & 0 & 0 \end{bmatrix}^T,\ \Psi_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & I & -I & 0 \end{bmatrix}^T,\\
\Psi_3 &= \begin{bmatrix} 0 & -I & 0 & 0 & I & 0 & 0 & 0 \end{bmatrix}^T,\ \Psi_4 = \begin{bmatrix} 0 & I & -I & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T,
\end{aligned}
$$

and $\xi(k) = \big[x^T(k) \;\; y^T(k-\tau(k)) \;\; y^T(k-\tau_M) \;\; f^T(y(k-\tau(k))) \;\; y^T(k) \;\; x^T(k-h(k)) \;\; x^T(k-h_M) \;\; g^T(x(k-h(k)))\big]^T$. By condition (4), it follows that there exist diagonally matrices $D_i \geq 0, (i = 1, 2)$ such that

$$r^k\{y^T(t-\tau(t))D_1y(t-\tau(t))) - f^T(y(t-\tau(t)))L_1^{-1}D_1L_1^{-1}f(y(t-\tau(t)))\} \geq 0, \tag{16}$$

$$r^k\{x^T(t-h(t))D_2x(t-h(t))) - g^T(x(t-h(t)))L_2^{-1}D_2L_2^{-1}g(x(t-h(t)))\} \geq 0, \tag{17}$$

where $L_1 = \mathrm{diag}\{l_{11}, \cdots, l_{1m}\}, L_2 = \mathrm{diag}\{l_{21}, \cdots, l_{2n}\}$. From the A1 and (5), it follows that

$$\sigma_1^T(rP_1 + r^{h_M}h_MZ_1)\sigma_1 \leq \lambda_1[\rho_1x^T(k)x(k) + \rho_2y^T(k-\tau(k)))y(k-\tau(k)))], \tag{18}$$

$$\sigma_2^T(rP_2 + r^{\tau_M}\tau_MZ_2)\sigma_2 \leq \lambda_2[\rho_3y^T(k)y(k) + \rho_4x^T(k-h(k)))x(k-h(k)))]. \tag{19}$$

Substituting (12)-(15) into (11), adding the left sides of (16)-(19) to $\mathbb{E}(V(k))$, and combining with (8)-(11), then we can obtain

$$\begin{aligned}
\mathbb{E}\big\{\triangle V(k)\big\} \leq & r^k\mathbb{E}\big\{\xi^T(k)\big[\Omega + h(k)LZ_1^{-1}L^T + (h_M - h(k))MZ_1^{-1}M^T \\
& + \tau(k)NZ_2^{-1}N^T + (\tau_M - \tau(k))TZ_2^{-1}T^T\big]\xi(k)\big\} \\
= & r^k\mu_1(k)\mathbb{E}\big\{\xi^T(k)\big[\Omega + h_MLZ_1^{-1}L^T + \tau_MNZ_2^{-1}N^T\big]\xi(k)\big\} \\
& + r^k\mu_2(k)\mathbb{E}\big\{\xi^T(k)\big[\Omega + h_MLZ_1^{-1}L^T + \tau_mNZ_2^{-1}N^T \\
& + (\tau_M - \tau_m)TZ_2^{-1}T^T\big]\xi(k)\big\} + r^k\mu_3(k)\mathbb{E}\big\{\xi^T(k)\big[\Omega \\
& + h_mLZ_1^{-1}L^T + (h_M - h_m)MZ_1^{-1}M^T + \tau_MNZ_2^{-1}N^T\big]\xi(k)\big\} \\
& + r^k\mu_4(k)\mathbb{E}\big\{\xi^T(k)\big[\Omega + h_mLZ_1^{-1}L^T + (h_M - h_m)MZ_1^{-1}M^T \\
& + \tau_mNZ_2^{-1}N^T + (\tau_M - \tau_m)TZ_2^{-1}T^T\big]\xi(k)\big\},
\end{aligned} \tag{20}$$

where Ω is defined in Theorem 1, and $\mu_1(k) = \frac{(h(k)-h_m)(\tau(k)-\tau_m)}{(h_M-h_m)(\tau_M-\tau_m)}, \mu_2(k) = \frac{(h(k)-h_m)(\tau_M-\tau(k))}{(h_M-h_m)(\tau_M-\tau_m)}, \mu_3(k) = \frac{(\tau(k)-\tau_m)(h_M-h(k))}{(h_M-h_m)(\tau_M-\tau_m)}$, $\mu_4(k) = \frac{(h_M-h(k))(\tau_M-\tau(k))}{(h_M-h_m)(\tau_M-\tau_m)}$. Therefore, if $\Omega+h_MLZ_1^{-1}L^T+\tau_MNZ_2^{-1}N^T < 0, \Omega+h_MLZ_1^{-1}L^T+\tau_mNZ_2^{-1}N^T+(\tau_M-\tau_m)TZ_2^{-1}T^T < 0, \Omega+h_mLZ_1^{-1}L^T+(h_M-h_m)MZ_1^{-1}M^T+\tau_MNZ_2^{-1}N^T < 0, \Omega+h_mLZ_1^{-1}L^T+(h_M-h_m)MZ_1^{-1}M^T+\tau_mNZ_2^{-1}N^T+(\tau_M-\tau_m)TZ_2^{-1}T^T < 0$, then we have $\mathbb{E}\{\Delta V(k)\} < 0$, which implies that

$$\mathbb{E}\{V(k)\} \leq \mathbb{E}\{V(k-1)\} \leq \mathbb{E}\{V(k-2)\} \leq \cdots \leq \mathbb{E}\{V(1)\} \leq \mathbb{E}\{V(0)\}. \tag{21}$$

By Schur complement, LMIs in (6) are equivalent to above matrix inequalities. Thus, if (5)-(6) hold, we can obtain $\mathbb{E}\{V(k)\} \leq \mathbb{E}\{V(0)\}$. Using the similar analysis as in [7], we can obtain

$$\mathbb{E}\{V(0)\} \leq \mu_1 \sup_{s\in\mathbb{N}[-h_M,0]} \mathbb{E}\{\|\phi(s)\|^2\} + \mu_2 \sup_{s\in\mathbb{N}[-\tau_M,0]} \mathbb{E}\{\|\psi(s)\|^2\}, \tag{22}$$

where $\mu_1 = \lambda_M(P_1)+[(h_M-h_m+1)\lambda_M(Q_1)+\lambda_M(R_1)+4r^{h_M}h_M\lambda_M(Z_1)]\frac{1-r^{-h_M}}{r-1}$, $\mu_2 = \lambda_M(P_2)+[(\tau_M-\tau_m+1)\lambda_M(Q_2)+\lambda_M(R_2)+4r^{\tau_M}\tau_M\lambda_M(Z_2)]\frac{1-r^{-\tau_M}}{r-1}$. On the other hand,

$$\mathbb{E}\{V(k)\} \geq r^k\lambda_m(P_1)\mathbb{E}\{\|x(k)\|^2\} + r^k\lambda_m(P_2)\mathbb{E}\{\|y(k)\|^2\}. \tag{23}$$

From (22) and (23), we can obtain: $\mathbb{E}\{\|x(k)\|^2\} + \mathbb{E}\{\|y(k)\|^2\}$ $\leq \frac{\alpha}{\beta}\big(\sup_{s\in\mathbb{N}[-h_M,0]}\mathbb{E}\{\|\phi(s)\|^2\} + \sup_{s\in\mathbb{N}[-\tau_M,0]}\mathbb{E}\{\|\psi(s)\|^2\}\big)r^{-k}$, where $\alpha = \max\{\mu_1,\mu_2\},\ \beta = \min\{\lambda_m(P_1),\lambda_m(P_2)\}$. Obviously, $\frac{\alpha}{\beta} > 1$, by Definition 1, the model (1) is globally exponentially stable in the mean square.

Based on the Theorem 1, we can easily obtain the exponential stability criterion of the following discrete-time delayed BAM neural network:

$$\begin{aligned} x(k+1) &= Ax(k) + Wf(y(k-\tau(k))), \\ y(k+1) &= By(k) + Vg(x(k-h(k))). \end{aligned} \tag{24}$$

Corollary 1 . The discrete-time delayed BAM neural network (24) is globally exponentially stable, if modified LMIs in (6) hold (ρ_i in Ω is fixed as 0).

4 Numerical Examples

Example 1. Consider the BAM NN (24) with the following parameters

$$A = \begin{bmatrix} 0.8 & 0 \\ 0 & 0.9 \end{bmatrix},\ B = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.4 \end{bmatrix},\ W = \begin{bmatrix} 0.1 & -0.01 \\ -0.2 & -0.1 \end{bmatrix},\ V = \begin{bmatrix} 0.15 & 0 \\ -0.2 & 0.1 \end{bmatrix}.$$

The activation functions satisfy Assumption A1 with $L_1 = L_2 = \mathrm{diag}\{1,1\}$.

By the Corollary 1 in this paper with $r = 1$, some maximum allowable delay bounds for guaranteeing the asymptotic stability of this system can be obtained. For a detailed comparison with the Theorem 1 in [5,7] and the Corollary 1 in [6], we made Table 1. For this example, it is obvious that the Corollary 1 in this paper is less conservative than the results in [5-7].

Table 1. Maximum allowable delay bounds for Example 1

$\tau_m = h_m$	2	4	6	8	10	15	20	25
$\tau_M = h_M$ [5, 6]	6	8	10	12	14	19	24	29
$\tau_M = h_M$ [7]	11	12	13	15	16	21	25	30
$\tau_M = h_M$ *Corollary* 1	13	14	15	16	17	21	26	30

Example 2. Consider the BAM NN (1) with the following parameters

$$A = \begin{bmatrix} 1/5 & 0 \\ 0 & 1/5 \end{bmatrix},\ B = \begin{bmatrix} 1/10 & 0 \\ 0 & 1/10 \end{bmatrix},\ W = \begin{bmatrix} 0 & 1/8 \\ 1/8 & 0 \end{bmatrix},\ V = \begin{bmatrix} -1/20 & 0 \\ 0 & -1/20 \end{bmatrix},$$

$\rho_1 = \rho_2 = \rho_3 = \rho_4 = 0.05,\ L_1 = L_2 = diag\{1,1\}$.

Assume that $h_m = \tau_m = 2, h_M = \tau_M = 5, r = 1.2$, we can find that LMIs (5)-(6) are feasible and part solutions are given as follows:

$P_1 = \text{diag}\{0.1310, 0.1310\}, P_2 = \text{diag}\{0.1486, 0.1486\}, Q_1 = \text{diag}\{0.0236, 0.0236\}$, $Q_2 = \text{diag}\{0.0309, 0.0309\}, R_1 = \text{diag}\{0.0041, 0.0041\}, R_2 = \text{diag}\{0.0043, 0.0043\}$.

By Theorem 1, it is concluded that BAM NN (1) with above parameters is global exponential stability in the mean square.

5 Conclusion

This paper considers the global exponential stability for discrete-time stochastic BAM neural networks with time-varying delays. Based on the Lyapunov stability theory and LMI technique, the delay-dependent exponential stability criteria are established . The obtained results are expressed by LMIs and can be checked by using the Matlab LMI Toolbox. Finally, we present two numerical examples to show the feasibility and less conservativeness of obtained results.

References

1. Arik, S.: Global Asymptotic Stability Analysis of Bidirectional Associative Memory Neural Networks with Time Delays. IEEE Trans. Neural Netw. 16, 580–586 (2005)
2. Park, J.H.: A Novel Criterion for Global Asymptotic Stability of BAM Neural Networks with Time Delays. Chaos, Solitons & Fractals 29, 446–453 (2006)
3. Cao, J., Ho, D.W.C., Huang, X.: LMI-Based Criteria for Global Robust Stability of Bidirectional Associative Memory Networks with Time Delay. Nonlinear Anal.: Real World Appl. 66, 1558–1572 (2007)
4. Liang, J., Cao, J., Ho, D.W.C.: Discrete-Time Bidirectional Associative Memory Neural Networks with Variable Delays. Phys. Lett. A 335, 226–234 (2005)
5. Liu, X.G., Tang, M.L., Martin, R., Liu, X.B.: Discrete-Time BAM Neural Networks with Variable Delays. Phys. Lett. A 367, 322–330 (2007)
6. Gao, M., Cui, B.: Global Robust Exponential Stability of Discrete-Time Interval BAM Neural Networks with Time-Varying Delays. Appl. Math. Model. 33, 1270–1284 (2009)
7. Chen, Y., Bi, W., Wu, Y.: Delay-Dependent Exponential Stability for Discrete-Time BAM Neural Networks with Time-Varying Delays. Discrete Dynamics in Nature and Society, Article ID 421614, 14 (2008), doi:10.1155/2008/421614
8. Blythe, S., Mao, X., Liao, X.: Stability of Stochastic Delay Neural Networks. J. Franklin Inst. 338, 481–495 (2001)
9. Huang, H., Feng, G.: Delay-Dependent Stability for Uncertain Stochastic Neural Networks with Time-Varying Selay. Physica A: Stat. Mech. Appl. 381, 93–103 (2007)
10. Liu, Y., Wang, Z., Liu, X.: Robust Stability of Discrete-Time Stochastic Neural Networks with Time-Varying Delays. Neurocomputing 71, 823–833 (2008)

Invariant and Globally Exponentially Attractive Sets of Separated Variables Systems with Time-Varying Delays

Zhengwen Tu, Jigui Jian, and Baoxian Wang

Institute of Nonlinear and Complex Systems, China Three Gorges University, Yichang, Hubei 443002, China
tuzhengwen@163.com, jiguijian@ctgu.edu.cn, baoxianwang2005@yahoo.com.cn

Abstract. In this paper, we study the positive invariant set, and globally exponentially attractive set for a class of nonlinear separated variables systems with bounded time-varying delays. Based on inequality techniques, the properties of non-negative matrices and vector Lyapunov function method, some algebraic criterions for the the above-mentioned sets are obtained. Finally, one example is given to demonstrate our results.

Keywords: Nonlinear system, Time-varying delays, Inequality techniques, Invariant set, Globally exponentially attractive set.

1 Introduction

Differential systems with separated variables and time delays are not only commonly seen in automatic control and theory of neural networks, but also often-used systems, such as delayed Hopfield neural networks model, delayed BAM networks model, delayed Lurie networks model and so on. Some authors have investigated the global stability of equilibrium point of separated variables systems[1,2]. Jian *et al.*[3] discussed the globally exponential stability of equilibrium states of a class of nonlinear separated variables systems with bounded time-varying delays by employing the Lyapunov function method and Halanay' delay differential inequality. However, the invariant set and the attractive set of the systems are rarely studied, leave alone their exponentially attractive set. In many applied subjects, we find that they play a great role to find periodic solutions and singular attractors. Motivated by the above discussions, and resorting to the ideas of [4,5], this paper is devoted to the problem of the positive invariant set and the globally exponentially attractive set of the nonlinear systems, and gives some sufficient conditions for the above-mentioned sets by the differential inequality techniques, the properties of non-negative matrices and vector Lyapunov function method.

The remaining paper is organized as follows: Section 2 describes some preliminaries including some necessary notations, definitions, assumptions and a lemma. The main results are stated in Section 3. Section 4 gives one numerical example to verify our main results. Finally, conclusions are made in Section 5.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 667–674, 2010.

2 Preliminaries

In this paper, R^+ means $R^+=[0,\infty)$, R^n denotes the n-dimensional Euclidean space, $C[X,Y]$ is a class of continuous mapping set from the topological space X to topological space Y. Especially, $\mathrm{C}\triangleq\mathrm{C}([-\tau,0],\mathrm{R}^n)$, where $\tau>0$.

Consider the nonlinear separated variables systems with n-dimensional

$$\frac{dx_i(t)}{dt}=\sum_{j=1}^{n}a_{ij}(t)f_j(x_j(t))+\sum_{j=1}^{n}b_{ij}(t)f_j(x_j(t-\tau_j(t))), \tag{1}$$

where $i,j=1,2,\cdots,n$, $0\le\tau_j(t)\le\tau$, τ is a positive constant, $a_{ij}(t)$ and $b_{ij}(t)$ are bounded and continuous functions on $[t_0-\tau,\infty)$, $f_j(0)=0, f_j(t)\in C[R,R]$. The initial conditions associated with (1) is of the form $x_i(t)=\varphi_i(t)$, $t\in[t_0-\tau,t_0]$, where φ_i is bounded and continuous on $[t_0-\tau,t_0]$. For any $\varphi=(\varphi_1,\ldots,\varphi_n)^T\in C$, a function $x=(x_1,\cdots,x_n)^T:[t_0-\tau,+\infty)\to R^n$ is the solution of the system (1) if it satisfies (1). And an equilibrium point of the system (1) is a constant vector $x^*=(x_1^*,x_2^*,\cdots,x_n^*)^T$ satisfying $\sum_{j=1}^{n}(a_{ij}(t_0)+b_{ij}(t_0))f_j(x_j^*)=0, i=1,2,\cdots,n$. Throughout the paper, we always assume that the system (1) has a continuous solution denoted by $x(t,t_0,\varphi)$ or simply $x(t)$ if no confusion should occur.

For convenience, we introduce some notations. We let I denotes identity matrix, $E=[1,1,\cdots,1]^T\in R^n$, and for any matrixes or vectors A and B, $A\ge B(A<B)$ means that each pair of corresponding elements of A and B satisfies the inequality $a_{ij}\ge b_{ij}(a_{ij}<b_{ij})$. Especially, A is called a nonnegative matrix if $A\ge 0$. For all $x\in R^n$, we define $[x(t)]^+=[|x_1(t)|,\cdots,|x_n(t)|]^T$, for $x\in C$, $[x(t)]_\tau^+=[|x_1(t)|_\tau,\cdots,|x_n(t)|_\tau]^T$, where

$$|x_i(t)|_\tau=\sup_{-\tau\le s\le 0}|x_i(t+s)|,\qquad i=1,\cdots,n.$$

Definition 2.1 ([4]). The set $S\subset C$ is called a positive invariant set of (1) if for any initial value $\varphi\in S$, we have $x(t,t_0,\varphi)\in S$.

Definition 2.2 ([4]). A set $S\subset C$ is called an attractive set of (1), if S possesses an open neighborhood D such that for any initial value $\varphi\in D$, the solution $x(t,t_0,\varphi)$ converges to S as $t\to+\infty$, that is

$$dist(x(t,t_0,\varphi),S)\to 0,\quad t\to+\infty,$$

where $dist(x,S)=\inf_{y\in S}d(x,y)$, and $d(x,y)$ denotes the distance of x to y in R^n.

The set S is called a global attractive set of (1) if $D=C$.

Definition 2.3 ([6]). The set $S=\{x\in R^n\mid V_i(x_i(t))\le\ell_i\}$ is called a globally exponentially attractive set of (1), if $V_i(x_i(t))$ is an radially unbounded and positive define Lyapunov function, there exist positive constants ℓ_i and λ such

that for any solution $x(t) = x(t, t_0, \varphi)$ of (1), $V_i(\varphi_i) > \ell_i$, $V_i(x_i(t)) > \ell_i$, $t \geq t_0$, implies

$$V_i(x_i(t)) - \ell_i \leq exp\{-\lambda(t - t_0)\}(\bar{V}_i(\varphi_i) - \ell_i), i = 1, 2, \cdots, n,$$

where $\bar{V}_i(\varphi_i) \geq V_i(\varphi_i)$ and $\bar{V}_i(\varphi_i)$ is a constant.

Remark. In fact, letting$K_i(\varphi) = \bar{V}_i(\varphi) - \ell_i$ in the Definition 2.4 of [5], we can obtain the Definition 2.3 of this paper.

Lemma ([8]). If $M \geq 0$ and $\rho(M) < 1$, then $(1-M)^{-1} \geq 0$, where $\rho(M)$ denotes the spectral radius of a square matrix M.

3 Main Results

Throughout this paper we always suppose that:

(H_1) $0 < \tilde{l}_i \leq \frac{f_i(x_i)}{x_i} \leq l_i$, $i = 1, 2, \cdots, n$, for any $x_i \in R \setminus \{0\}$;
(H_2) $a_{ii}(t) \leq a_{ii} < 0$, $|a_{ij}(t)| \leq a_{ij}$, for $i \neq j$, and $|b_{ij}(t)| \leq b_{ij}$, $i, j = 1, 2, \cdots, n$;
(H_3) $\rho(M) < 1$.

Where $M = Q^{-1}(A^*L + BL)$, $L = diag\{l_1, l_2, \cdots, l_n\}$, $B = (b_{ij})_{n\times n}$, $q_i = -a_{ii}\tilde{l}_i$, $i = 1, 2, \cdots, n$, $Q = diag\{q_1, q_2, \ldots, q_n\}$, $Q^{-1} = diag\{\frac{1}{q_1}, \frac{1}{q_2}, \cdots, \frac{1}{q_n}\}$, $A^* = [(1 - \delta_{ij})a_{ij}]_{n\times n}$, and $\delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$

Theorem 3.1. If the hypotheses $(H_1) \sim (H_3)$ hold, then the set

$$S = \{\varphi \in C \mid [\varphi]_\tau^+ \leq K = (I - M)^{-1}E\alpha\}$$

is a positive invariant set of the system(1), where α is an arbitrary positive constant.

Proof. We first prove, for any given $\beta > 1$, when $[\varphi]_\tau^+ < \beta K$,

$$[x(t)]^+ < \beta K, \quad t \geq t_0. \tag{2}$$

If it is not true, there must be some i, and $t_1 > t_0$, such that

$$[x_i(t_1)] = \beta k_i, \tag{3}$$

and

$$[x_i(t_1)] \leq \beta k_i, \quad t_0 \leq t \leq t_1, \tag{4}$$

where k_i is the ith component of vector K.

From (1) and $(H_1) \sim (H_3)$, one yields

$$\begin{aligned} D^+|x_i(t)|\,|_{(1)} &= a_{ii}(t)f_i(x_i(t))\text{sgn } x_i(t) + \sum_{j=1,j\neq i}^{n} a_{ij}(t)f_j(x_j(t))\text{sgn } x_i(t) \\ &\quad + \sum_{j=1}^{n} b_{ij}(t)f_j(x_j(t-\tau_j(t)))\text{sgn } x_i(t) \\ &\leq a_{ii}\tilde{l}_i|x_i(t)| + \sum_{j=1,j\neq i}^{n} l_j a_{ij}|x_j(t)| + \sum_{j=1}^{n} l_j b_{ij}|x_j(t)|_\tau,\ t > t_0, \end{aligned}$$

then

$$D^+|x_i(t)|\,|_{(1)} \leq -Q[x(t)]^+ + A^*L[x(t)]^+ + BL[x[t-\tau(t)]_\tau^+,\ t > t_0.$$

It follows that

$$\begin{aligned} [x(t_1)]^+ &\leq e^{-Q(t_1-t_0)}[\varphi]_\tau^+ + \int_{t_0}^{t_1} e^{-Q(t_1-s)}\{A^*L[x(s)]^+ + BL[x(s)]_\tau^+\}\mathrm{d}s \\ &\leq e^{-Q(t_1-t_0)}\beta K + \int_{t_0}^{t_1} e^{-Q(t_1-s)}\{A^*L\beta K + BL\beta K\}\mathrm{d}s \\ &= e^{-Q(t_1-t_0)}\beta K + \int_{t_0}^{t_1} e^{-Q(t_1-s)}QM\beta K\mathrm{d}s \\ &= e^{-Q(t_1-t_0)}\beta K + (I - e^{-Q(t_1-t_0)})M\beta K \\ &= e^{-Q(t_1-t_0)}(\beta K - M\beta K) + M\beta K. \end{aligned}$$

Noting that $\beta > 1$ and $\beta K = \beta(MK + E\alpha) > \beta MK$, so we obtain

$$[x(t_1)]^+ \leq e^{-Q(t_1-t_0)}(\beta K - M\beta K) + M\beta K < \beta K,$$

which contradicts (3), and so (2) holds. Letting $\beta \to 1$ in (3), when $[\varphi]_\tau^+ \leq K$, we have for all $\varphi \in C$,

$$[x(t)]^+ \leq K, \quad t \geq t_0. \tag{5}$$

Following from Definition 2.1, the Theorem can be proved.

Corollary 3.1. If the system (1) has an equilibrium point and the hypotheses $(H_1) \sim (H_3)$ hold, then the equilibrium point of is uniformly stable.

From the proof of Theorem 3.1 and (2), for a given $\varphi \in C$, there exists $\beta > 1$, for all $t \geq t_0$, when $[\varphi]_\tau^+ < \beta K$, then $[x(t)]^+ < \beta K$, so we have the following Corollary.

Corollary 3.2. If the hypotheses $(H_1) \sim (H_3)$ hold, then the solutions of (1) are uniformly bounded.

Theorem 3.2. If the hypotheses $(H_1) \sim (H_3)$ hold, then the set

$$S = \{\varphi \in C \mid [\varphi]_\tau^+ \leq K = (I - M)^{-1} E\alpha\}$$

is an attractive set of the system (1), where α is an arbitrary positive constant.

Proof. For any $\varphi \in C$, we have the solution $x(t)$ satisfies

$$\lim_{t \to +\infty} \sup [x(t)]^+ = K, \tag{6}$$

otherwise, from (5), we know that there must be a constant vector $\sigma > 0$ such that

$$\lim_{t \to +\infty} \sup [x(t)]^+ = K + \sigma. \tag{7}$$

According to the definition of superior limit, for sufficient small constant $\varepsilon > 0$, there exists $t_1 > t_0$ such that

$$[x(t - \tau(t))]^+ \leq K + \sigma + \varepsilon E. \tag{8}$$

Because of $Q = diag\{q_1, \cdots, q_n\}$ with $q_i > 0$, $i = 1, \cdots, n$, for the above $\varepsilon > 0$, there is $T > 0$, such that

$$\int_T^\infty e^{-Qs} QM\beta K \mathsf{d}s < \varepsilon E. \tag{9}$$

From (1), (2), (8) and (9), when $t \geq t_2 + T$, we obtain

$$\begin{aligned}
[x(t)]^+ &\leq e^{-Q(t-t_0)}[\varphi]_\tau^+ + \int_{t_0}^t e^{-Q(t-s)}\{A^* L[x(s)]^+ + BL[x(s)]_\tau^+\}\mathsf{d}s \\
&= e^{-Q(t-t_0)}[\varphi]_\tau^+ + (\int_{t_0}^{t-T} + \int_{t-T}^t) e^{-Q(t-s)}\{A^* L[x(s)]^+ + BL[x(s)]_\tau^+\}\mathsf{d}s \\
&\leq e^{-Q(t-t_0)}[\varphi]_\tau^+ + \int_T^{+\infty} e^{-Qs} QM\beta K \mathsf{d}s \\
&\quad + \int_{t-T}^t e^{-Q(t-s)} QM(K + \sigma + \varepsilon E)\}\mathsf{d}s \\
&\leq e^{-Q(t-t_0)}[\varphi]_\tau^+ + \varepsilon E + MK + M\sigma + \varepsilon ME.
\end{aligned}$$

Combining (7) with the definition of superior limit, there exists $t_w \geq t_2 + T$, $w = 1, 2, \cdots$, such that

$$\lim_{t_w \to \infty} [x(t_w)]^+ = K + \sigma.$$

Letting $t_w \to +\infty$, $\varepsilon \to 0$ and noting $K = MK + \alpha E$, we obtain

$$\sigma + K \leq MK + M\sigma < K + M\sigma,$$

i.e. $\sigma < M\sigma$, then from Theorem 8.3.2 of [8], $\sigma \geq 0$ and σ is not always zero, one has $\rho(M) \geq 1$, which contradicts $\rho(M) < 1$. Hence, σ must be zero vector and (6) holds. The proof is finished.

Corollary 3.3. If the hypotheses $(H_1) \sim (H_3)$ hold, then the solutions of (1) are globally asymptotically stable.

Theorem 3.3. If the hypotheses $(H_1) \sim (H_3)$ hold, then the set

$$S^* = \{x(t) \mid |x_i(t)| \leq \frac{1}{\lambda}[\sum_{j=1, j\neq i}^{n} a_{ij} l_j \beta k_j + \sum_{j=1}^{n} b_{ij} l_j \beta k_j] \triangleq \tilde{k}_i\}$$

is a globally exponentially attractive set of the system (1), where $0 < \lambda \leq \min\limits_{1\leq i\leq n}\{q_i\}$, $\beta > 1$, and k_i is the ith component of vector K which is given in Theorem 3.1.

Proof. We consider the radially unbounded and positive define Lypunov function

$$V(x(t)) = (V_1, V_2, \ldots, V_n)^T = (|x_1(t)|, |x_2(t)|, \cdots, |x_n(t)|)^T.$$

Choose λ such that $0 < \lambda \leq \min\limits_{1\leq i\leq n}\{q_i\}$, and let $\gamma = (\gamma_1, \gamma_2, \cdots, \gamma_n)^T$, where $\gamma_i = \sum\limits_{j=1, j\neq i}^{n} a_{ij} l_j \beta k_j + \sum\limits_{j=1}^{n} b_{ij} l_j \beta k_j$. We consider another radially unbounded and positive definite Lypunov function $W(t) = e^{\lambda t}[V(x(t)) - \frac{\gamma}{\lambda}]$, and $W_i(t)$ denotes the ith component of vector $W(t)$, then

$$\begin{aligned}
D^+ W_i(t) \mid_{(1)} &= \lambda e^{\lambda t}|x_i(t)| + e^{\lambda t}\frac{x_i(t)}{dt}\text{sgn } x_i(t) - \gamma_i e^{\lambda t} \\
&= e^{\lambda t}[\lambda |x_i(t)| + \sum_{j=1}^{n} a_{ij}(t) f_j(x_j(t))\text{sgn } x_i(t) \\
&\quad + \sum_{j=1}^{n} b_{ij}(t) f_j(x_j(t-\tau_j(t)))\text{sgn } x_i(t) - \gamma_i] \\
&\leq e^{\lambda t}[\lambda |x_i(t)| + a_{ii}\tilde{l}_i |x_i(t)| + \sum_{j=1, j\neq i}^{n} l_j a_{ij} |x_j(t)| \\
&\quad + \sum_{j=1}^{n} l_j b_{ij} |x_j(t)|_\tau - \gamma_i],
\end{aligned}$$

From Corollary 3.2, we obtain

$$\begin{aligned}
D^+ W_i(t) \mid_{(1)} &= e^{\lambda t}[-(q_i - \lambda)|x_i(t)| + \sum_{j=1, j\neq i}^{n} l_j |a_{ij}| \beta k_j \\
&\quad + \sum_{j=1}^{n} l_j |b_{ij}| \beta k_j - \gamma_i] \\
&= -e^{-\lambda t}(q_i - \lambda)|x_i(t)| \leq 0.
\end{aligned}$$

that is

$$D^+ e^{\lambda} V_i \mid_{(1)} \leq 0. \tag{10}$$

Integrating two sides of (10) from t_0 to arbitrary $t > t_0$, we have

$$e^{\lambda t}[V_i(x(t)) - \frac{\gamma_i}{\lambda}] \leq V_i(x(t_0)) - \frac{\gamma_i}{\lambda},$$

i.e.

$$V_i(x(t)) - \frac{\gamma_i}{\lambda} \leq e^{-\lambda t}[V_i(x(t_0)) - \frac{\gamma_i}{\lambda}].$$

Following from Definition 2.3, the proof is completed.

4 An Illustrative Example

Example. Consider the following example:

$$\begin{cases} \frac{dx_1(t)}{dt} = -5f_1(x_1(t)) + 0.1f_1(x_1(t-\tau_1(t))) + 0.05f_2(x_2(t-\tau_2(t))) \\ \frac{dx_2(t)}{dt} = -5f_2(x_2(t)) + 0.05f_1(x_1(t-\tau_1(t))) - 0.1f_2(x_2(t-\tau_2(t))) \end{cases} \tag{11}$$

where $f_i(x_i(t)) = \frac{sinx_i(t)}{3} + \frac{x_i(t)}{3}$ $(i = 1, 2)$, $\frac{1}{3} \leq \frac{f_i(x_i)}{x_i} \leq 1$, we choose $\tilde{l}_1 = \tilde{l}_2 = \frac{1}{3}$, $l_1 = l_2 = 1$, by simply calculation, we obtain

$$L = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},\ Q = \begin{pmatrix} \frac{5}{3} & 0 \\ 0 & \frac{5}{3} \end{pmatrix},\ Q = \begin{pmatrix} \frac{3}{50} & \frac{3}{100} \\ \frac{3}{100} & \frac{3}{50} \end{pmatrix},\ (I-M)^{-1} = \begin{pmatrix} \frac{9400}{8827} & \frac{300}{8827} \\ \frac{300}{8827} & \frac{9400}{8827} \end{pmatrix}.$$

And choosing $\alpha = \frac{8827}{10000}$, we have $K = (\frac{97}{100}, \frac{97}{100})^T$. According to Theorem 3.1 and Theorem 3.2, we can know that the set $S = \{\varphi \in C \mid |\varphi_1| \leq \frac{97}{100}, |\varphi_2| \leq \frac{97}{100}\}$ is the invariant set and attractive set of the system (11). And we let $\lambda = \frac{5}{4}$, $\beta = \frac{5}{3}$, from Theorem 3.3, we obtain that the set $S^* = \{x(t) \mid |x_1(t)| \leq \frac{97}{500}, |x_2(t)| \leq \frac{97}{500}\}$ is a globally exponentially attractive set of the system (11).

5 Conclusion

This paper has established detailed estimations of the positive invariant set and the globally exponentially attractive set together with the attractive set of a class nonlinear separated variables systems with bounded time-varying delays. One example has been worked out, and it implies that the attractive set is larger than the exponentially attractive set, which confirms the validity of our results.

Acknowledgments

This work is supported by the Graduate Scientific Research Creative Foundation of Three Gorges University (200946)(200945), the National Natural

Science Foundation, China (60974136), the Scientific Innovation TeamProject of Hubei Provincial Department of Education (T200809), and Hubei Natural Science Foundation (2008CDB316) (2008CDZ046).

References

1. Liao, X.X.: Mathematic Theory and Appliations of Stability, 2nd edn. Huazhong Normal University Press, Wuhan (2001)
2. Liu, B.Y., Gui, W.H.: Stabilization controller for a Class of Nonlinear Discrete Control Systems with SeparatedVariables. Mathematical Theory and Applications 23(3), 53–56 (2003)
3. Jian, J.G., Kong, D.M., Luo, H.G., Liao, X.X.: Exponential Stability of Differential Systems with Separated Variables and Time Delays. Journal of Central South University 36(2), 282–287 (2005)
4. Xu, D.Y., Zhao, H.Y.: Invariant set and attractivity of nonlinear differential equations with delays. Journal of Applied Mathematics 15(3), 321–325 (2002)
5. Zhao, H.Y.: Invairant and attractor of nonautonomous functional differential systems. Mathematical analysis and applications 282, 437–443 (2003)
6. Liao, X.X., Luo, Q., Zeng, Z.G., Guo, Y.X.: Global exponential stability in Lagrange sense for recurrent neural networks with time delays. Nonlinear Analysis: Real World Applications 9, 1535–1557 (2008)
7. Liao, X.X., Wang, J.: Global dissipativity of continuous-time recurrent neural networks with time delay. Physical Review E 68, 1–7 (2003)
8. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1985)

Delay-Dependent Stability of Nonlinear Uncertain Stochastic Systems with Time-Varying Delays

Cheng Wang[1,2]

[1] College of Mathematics and Information Science, Huanggang Normal University, Huanggang 438000, China
[2] Institute of Systems Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
wangc80@163.com

Abstract. This paper considers the problem of delay-dependent stability of the systems with nonlinearity, uncertainty and time-varying delays. The uncertainty is assumed to be of norm-bounded form. By constructing Lyapunov-Krasovskii functional and introducing appropriate free-weighting matrices, the sufficient delay-dependent condition is derived for the asymptotic stability of the system. The proposed result is formulated in terms of linear matrix inequality, which can be efficiently solved by standard convex optimization algorithms.

Keywords: Stochastic system, Delay-dependent, Stability, Linear matrix inequality (LMI).

1 Introduction

During the past decades, the problem of robust stability analysis of delay systems has received significant attention. Time-delays are frequently encountered in practical systems such as engineering, communications and biological systems and may induce instability, oscillation and poor performance. The stability and control of time-delay systems have been of great importance and interest. Current efforts can be classified into two catalogs according to their dependence on the information about the size of time-delays of the system, namely delay-independent stability criterion (see [1-3]) and delay-dependent stability criterion (see [4-6]). The delay-independent stability is independent of the size of the delays and delay-dependent stability is concerned with the size of delays. Generally, for the cases of small delays, delay-independent results are more conservative than those dependent on the size of delays. Therefore, increasing attention has been focused on the delay-dependent stability of stochastic systems recently. For example, in [7], the problem of delay-dependent stability and delayed-state-feedback stabilization of uncertain stochastic systems were studied, and a delay-dependent criterion for exponential stability of uncertain stochastic delay systems was established. In [8], based on a neutral transformation with Markovian switching and linear matrix inequality (LMI) technique, some new delay-dependent stability criteria were derived for uncertain stochastic systems with time-delays and Markovian jump parameters. In [9], by using a descriptor model transformation and applying Moon's

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 675–683, 2010.

inequality for bounding cross terms, delay-dependent exponential stability criterion for stochastic systems with multiple delays was proposed. However, in the existing literatures, most of them were about the stability for uncertain linear stochastic systems. The problem of delay-dependent stability for nonlinear uncertain stochastic systems with time-varying delays has not been fully investigated, which still remains an interesting research topic.

In this paper, we aim to solve the problem of delay-dependent stability for nonlinear uncertain stochastic systems with time-varying delays. Based on *Itô* differential formula and Lyapunov stability theory, sufficient delay-dependent criterion is derived in terms of LMI. Instead of employing the traditional model transformation, we introduce appropriate free-weighting matrices that can be chosen properly to lead to a less conservative result. The condition can be easily checked by resorting to available software packages.

Notation. The superscript "T" stands for matrix transposition. R^n denotes the n-dimensional Euclidean space. $R^{n\times m}$ is the set of all real matrices of dimension $m\times n$. $P>0$ means that P is real symmetric and positive definite. The notation $|\cdot|$ refers to the Euclidean vector norm in R^n, and $\|\cdot\|$ stands for the Euclidean norm for vector or the spectral norm of matrices. $E\{\cdot\}$ stands for the mathematical expectation. The symbol $*$ is used to denote a matrix which can be inferred by symmetry.

2 Problem Formulation

Consider the following stochastic system with nonlinearity, uncertainty and time-varying delays:

$$\begin{cases} dx(t) = [A(t)x(t) + B(t)x(t-\tau(t)) + f(t,x(t),x(t-\tau(t))]dt \\ \qquad\quad + [C(t)x(t) + D(t)x(t-\tau(t))]dw(t) \\ x(t) = \varphi(t), t\in[-h,0] \end{cases} \tag{1}$$

where $x(t)\in R^n$ is the state vector; $A(t)$, $B(t)$, $C(t)$ and $D(t)$ are matrix functions with time-varying uncertainties described as follows

$$A(t) = A + \Delta A(t)\ , B(t) = B + \Delta B(t)\ , C(t) = C + \Delta C(t)\ , D(t) = D + \Delta D(t)\ ,$$

where A, B, C and D are known constant matrices while uncertainties $\Delta A(t)$, $\Delta B(t)$, $\Delta C(t)$ and $\Delta D(t)$ are assumed to be of the following form:

$$[\Delta A(t)\ \Delta B(t)\ \Delta C(t)\ \Delta D(t)] = HF(t)[E_1\ E_2\ E_3\ E_4] \tag{2}$$

where H, E_1, E_2, E_3, E_4 are known real constant matrices with appropriate dimensions, and $F(t)$ is the time-varying uncertain matrix satisfying $F^T(t)F(t)\le I$. Vector $\omega(t)$ is an m-dimensional Brownian motion defined on a probability space. $\varphi(t)$ is an initial condition. The time-delay of system $\tau(t)$ is a known function that

satisfies $0 \le \tau(t) \le h < \infty$, $\dot{\tau}(t) \le \tau$. $f(t, x(t), x(t-\tau(t)))$ denotes the nonlinear uncertainties which satisfies the following condition:

$$\| f(t, x(t), x(t-\tau(t))) \| \le \| F_1 x(t) \| + \| F_1 x(t-\tau(t)) \| \tag{3}$$

with known constant matrices of appropriate dimensions F_1 and F_2.

The purpose of this paper is to develop a delay-dependent stability criterion for stochastic system (1). For this purpose, the following definition is first introduced.

Definition 1. For all admissible uncertainties satisfying (2), the nonlinear uncertain stochastic delay system is said to be robustly stochastically stable, if for any $\varepsilon > 0$, there exist $\delta(\varepsilon) > 0$ such that $\sup_{-h \le t \le 0} \{E \mid \varphi(t) \mid\} < \delta(\varepsilon)$, then the following inequality holds:

$$E\{\| x(t) \|^2\} < \varepsilon$$

and is said to be stochastically asymptotically stable if for any initial condition, we have

$$\lim_{t \to \infty} E\{\| x(t) \|^2\} = 0$$

3 Main Results

Let us give the following lemmas which will be used in the proof of our main result in this paper.

Lemma 1[10]**.** (Schur complement) If the constant matrices $\Sigma_1, \Sigma_2, \Sigma_3$ where $\Sigma_1 = \Sigma_1^T$ and $0 < \Sigma_2 = \Sigma_2^T$, then $\Sigma_1 + \Sigma_3^T \Sigma_2^{-1} \Sigma_3 < 0$ if and only if $\begin{bmatrix} \Sigma_1 & \Sigma_3^T \\ \Sigma_3 & -\Sigma_2 \end{bmatrix} < 0$. or, equivalently $\begin{bmatrix} -\Sigma_2 & \Sigma_3 \\ \Sigma_3^T & \Sigma_1 \end{bmatrix} < 0$.

Lemma 2[11]**.** Given appropriately dimensioned matrices φ, M, N with $\varphi = \varphi^T$, Then,

$$\varphi + MF(t)N + N^T F^T(t) M^T < 0$$

holds for all $F(t)$ satisfying $F^T(t)F(t) \le I$ if and only of for some $\varepsilon > 0$

$$\varphi + \varepsilon N^T N + \varepsilon^{-1} M M^T < 0$$

Lemma 3[12]**.** Let X, Y be real matrices of appropriate dimensions. Then, for any scalar $\varepsilon > 0$ and vector $x, y \in R^n$, we have

$$2x^T XYy \le \varepsilon^{-1} x^T X^T Xx + \varepsilon y^T Y^T Yy$$

Lemma 4[13]**.** For any pair of symmetric positive definite constant matrix $R \in R^{n\times n}$, scalar $r > 0$, and a vector function $x:[0,t] \to R^n$, the following inequality holds:

$$-h\int_{t-h}^{t} x^T(s)Rx(s)ds \le \int_{t-h}^{t} x^T(s)ds \cdot R \cdot \int_{t-h}^{t} x(s)ds$$

We will use LMI method and free-weighting matrices to solve the asymptotical stability of the stochastic system (1). The following theorem presents a sufficient delay-dependent condition.

Theorem 1. Consider the system (1). For given scalar $h \ge 0$ and τ, the nonlinear stochastic uncertain system with time-varying delays is stochastically asymptotically stable if there exist matrices $P > 0$, $Q > 0$, $R > 0$. N_1, N_2, N_3, M_1, M_2, M_3, and scalar numbers $\varepsilon_i > 0\ (i = 1,2,3)$ and $\mu_i > 0\ (i = 1,2)$ such that

$$\Phi = \begin{bmatrix} \Phi_{11} & \Phi_{12} & \Phi_{13} & N_1 & C^TP & M_1H & 0 & M_1 & 0 & 0 \\ * & \Phi_{22} & \Phi_{23} & N_2 & D^TP & M_2H & 0 & 0 & M_2 & 0 \\ * & * & \Phi_{33} & N_3 & 0 & M_3H & 0 & 0 & 0 & M_3 \\ * & * & * & -R & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & -P & 0 & PH & 0 & 0 & 0 \\ * & * & * & * & * & -\mu_1 I & 0 & 0 & 0 & 0 \\ * & * & * & * & * & * & -\mu_2 I & 0 & 0 & 0 \\ * & * & * & * & * & * & * & -\varepsilon_1 I & 0 & 0 \\ * & * & * & * & * & * & * & * & -\varepsilon_2 I & 0 \\ * & * & * & * & * & * & * & * & * & -\varepsilon_3 I \end{bmatrix} < 0 \qquad (4)$$

Where

$$\Phi_{11} = M_1A + A^TM_1^T + Q - N_1 - N_1^T + 2(\varepsilon_1+\varepsilon_2+\varepsilon_3)F_1^TF_1 + \mu_1E_1^TE_1 + \mu_2E_3^TE_3$$

$$\Phi_{12} = M_1B + A^TM_2^T + N_1 - N_2^T + \mu_1E_1^TE_2 + \mu_2E_3^TE_4$$

$$\Phi_{13} = P + A^TM_3^T - M_1 - N_3^T$$

$$\Phi_{22} = M_2B + B^TM_2^T - (1-\tau)Q + N_2 + N_2^T + 2(\varepsilon_1+\varepsilon_2+\varepsilon_3)F_2^TF_2 + \mu_1E_2^TE_2 + \mu_2E_4^TE_4$$

$$\Phi_{23} = B^TM_3^T - M_2 + N_3^T,$$

$$\Phi_{33} = -M_3 - M_3^T + h^2R$$

Proof. For convenience, let

$$y(t) = A(t)x(t) + B(t)x(t-\tau(t)) + f(t,x(t),x(t-\tau(t))),$$

$$g(t) = C(t)x(t) + D(t)x(t-\tau(t)) \qquad (5)$$

Then system (1) becomes

$$dx(t) = y(t)dt + g(t)d\omega(t) \tag{6}$$

Choose a Lyapunov functional candidate to be

$$V(t) = x^T(t)Px(t) + \int_{t-\tau(t)}^{t} x^T(s)Qx(s)ds + h\int_{t-h}^{t}\int_{s}^{t} y^T(\alpha)Ry(\alpha)d\alpha ds \tag{7}$$

where P, Q and R are real symmetric positive-definite matrices to be determined. Then by making use of the *Itô* differential rule, we obtain the stochastic differential as

$$dV(t) = LV(t)dt + 2x^T(t)Pg(t)d\omega(t) \tag{8}$$

where

$$\begin{aligned} LV(t) &= 2x^T(t)Py(t) + x^T(t)Qx(t) - (1-\dot{\tau}(t))x^T(t-\tau(t))Qx(t-\tau(t)) \\ &\quad + g^T(t)Pg(t) + h^2 y^T(t)Ry(t) - h\int_{t-h}^{t} y^T(s)Ry(s)ds \end{aligned} \tag{9}$$

Integrating (6) from $t-\tau(t)$ to t gives

$$x(t) - x(t-\tau(t)) = \int_{t-\tau(t)}^{t} y(s)ds + \int_{t-\tau(t)}^{t} g(s)d\omega(s)$$

which yields for any matrices N_1, N_2 and N_3 with appropriate dimensions

$$\begin{aligned} &(x^T(t)N_1 + x^T(t-\tau(t))N_2 + y^T(t)N_3)[x(t-\tau(t)) - x(t) \\ &+ \int_{t-\tau(t)}^{t} y(s)ds + \int_{t-\tau(t)}^{t} g(s)d\omega(s)] = 0 \end{aligned} \tag{10}$$

Similarly, for any matrices M_1, M_2 and M_3 with appropriate dimensions, we have

$$\begin{aligned} &(x^T(t)M_1 + x^T(t-\tau(t))M_2 + y^T(t)M_3)[A(t)x(t) \\ &+ B(t)x(t-\tau(t)) + f(t,x(t),x(t-\tau(t))) - y(t)] = 0 \end{aligned} \tag{11}$$

By adding the left sides of (10), (11) to (8), we can obtain

$$\begin{aligned} dV(t) &= L\tilde{V}(t)dt + 2x^T(t)Pg(t)d\omega(t) \\ &\quad + 2(x^T(t)N_1 + x^T(t-\tau(t))N_2 + y^T(t)N_3)\times\int_{t-\tau(t)}^{t} g(s)d\omega(s) \end{aligned} \tag{12}$$

where

$$\begin{aligned} L\tilde{V}(t) &= 2x^T(t)Py(t) + x^T(t)Qx(t) - (1-\dot{\tau}(t))x^T(t-\tau(t))Qx(t-\tau(t)) \\ &\quad + x^T(t-\tau(t))D^T(t)PC(t)x(t) + x^T(t-\tau(t))D^T(t)PD(t)x(t-\tau(t)) \\ &\quad + x^T(t)C^T(t)PC(t)x(t) + x^T(t)C^T(t)PD(t)x(t-\tau(t)) \ + h^2 y^T(t)Ry(t) \\ &\quad - h\int_{t-h}^{t} y^T(s)Ry(s)ds \ + 2[x^T(t)N_1 + x^T(t-\tau(t))N_2 + y^T(t)N_3] \\ &\quad \times[x(t-\tau(t)) - x(t) + \int_{t-\tau(t)}^{t} y(s)ds] \ + 2[x^T(t)M_1 + x^T(t-\tau(t))M_2 \\ &\quad + y^T(t)M_3]\times[A(t)x(t) + B(t)x(t-\tau(t)) + f(t,x(t),x(t-\tau(t))) - y(t)] \end{aligned} \tag{13}$$

Noting (3) and using Lemma 3, we have

$$\begin{aligned}2x^T(t)M_1 f(t,x(t),x(t-\tau(t))) &\le \varepsilon_1^{-1}x^T(t)M_1M_1^Tx(t)\\ &\quad+\varepsilon_1 f^T(t,x(t),x(t-\tau(t)))f(t,x(t),x(t-\tau(t)))\\ &\le \varepsilon_1^{-1}x^T(t)M_1M_1^Tx(t)+2\varepsilon_1x^T(t)F_1^TF_1x(t)\\ &\quad+2\varepsilon_1x^T(t-\tau(t))F_2^TF_2x(t-\tau(t)) \end{aligned} \tag{14}$$

$$\begin{aligned}2x^T(t-\tau(t))M_2 f(t,x(t),x(t-\tau(t))) &\le \varepsilon_2^{-1}x^T(t-\tau(t))M_2M_2^Tx(t-\tau(t))\\ &\quad+\varepsilon_2 f^T(t,x(t),x(t-\tau(t)))f(t,x(t),x(t-\tau(t)))\\ &\le \varepsilon_2^{-1}x^T(t-\tau(t))M_2M_2^Tx(t-\tau(t))+2\varepsilon_2x^T(t)F_1^TF_1x(t)\\ &\quad+2\varepsilon_2x^T(t-\tau(t))F_2^TF_2x(t-\tau(t)) \end{aligned} \tag{15}$$

$$\begin{aligned}2y^T(t)M_3 f(t,x(t),x(t-\tau(t))) &\le \varepsilon_3^{-1}y^T(t)M_3M_3^Ty(t)\\ &\quad+\varepsilon_3 f^T(t,x(t),x(t-\tau(t)))f(t,x(t),x(t-\tau(t)))\\ &\le \varepsilon_3^{-1}y^T(t)M_3M_3^Ty(t))+2\varepsilon_3x^T(t)F_1^TF_1x(t)\\ &\quad+2\varepsilon_3x^T(t-\tau(t))F_2^TF_2x(t-\tau(t)) \end{aligned} \tag{16}$$

By Lemma 4 and inequalities (14)-(16), we see

$$\begin{aligned}L\tilde{V}(t) &\le 2x^T(t)Py(t)+x^T(t)Qx(t)-(1-\tau)x^T(t-\tau(t))Qx(t-\tau(t))\\ &\quad+x^T(t-\tau(t))D^T(t)PC(t)x(t)+x^T(t-\tau(t))D^T(t)PD(t)x(t-\tau(t))\\ &\quad+x^T(t)C^T(t)PC(t)x(t)+x^T(t)C^T(t)PD(t)x(t-\tau(t))\ +h^2y^T(t)Ry(t)\\ &\quad-\int_{t-h}^{t}y^T(s)ds\cdot R\cdot\int_{t-h}^{t}y(s)ds\ +2[x^T(t)N_1+x^T(t-\tau(t))N_2+y^T(t)N_3]\\ &\quad\times x(t-\tau(t))\ -2[x^T(t)N_1+x^T(t-\tau(t))N_2+y^T(t)N_3]\times x(t)\\ &\quad+2[x^T(t)N_1+x^T(t-\tau(t))N_2+y^T(t)N_3]\times\int_{t-\tau(t)}^{t}y(s)ds\\ &\quad+2[x^T(t)M_1+x^T(t-\tau(t))M_2+y^T(t)M_3]\times A(t)x(t)\\ &\quad+2[x^T(t)M_1+x^T(t-\tau(t))M_2+y^T(t)M_3]\times B(t)x(t-\tau(t))\\ &\quad+2[x^T(t)M_1+x^T(t-\tau(t))M_2+y^T(t)M_3]\times f(t,x(t),x(t-\tau(t)))\\ &\quad+\varepsilon_1^{-1}x^T(t)M_1M_1^Tx(t)+\varepsilon_2^{-1}x^T(t-\tau(t))M_2M_2^Tx(t-\tau(t))\\ &\quad+\varepsilon_3^{-1}y^T(t)M_3M_3^Ty(t)+(2\varepsilon_1+2\varepsilon_2+2\varepsilon_3)x^T(t)F_1^TF_1x(t)\\ &\quad+(2\varepsilon_1+2\varepsilon_2+2\varepsilon_3)x^T(t-\tau(t))F_2^TF_2x(t-\tau(t))\\ &\quad-2[x^T(t)M_1+x^T(t-\tau(t))M_2+y^T(t)M_3]\times y(t)\ =\xi^T(t)\Theta\xi(t)\end{aligned} \tag{17}$$

where

$$\xi^T(t)=[x^T(t)\ \ x^T(t-\tau(t))\ \ y^T(t)\ \ \int_{t-h}^{t}y^T(s)ds]$$

$$\Theta=\begin{bmatrix}\Theta_{11}+C^T(t)PC(t) & \Theta_{12}+C^T(t)PD(t) & \Theta_{13} & N_1\\ * & \Theta_{22}+D^T(t)PD(t) & \Theta_{23} & N_2\\ * & * & \Theta_{33} & N_3\\ * & * & * & -R\end{bmatrix}$$

with

$$\Theta_{11}=M_1A(t)+A^T(t)M_1^T+Q-N_1-N_1^T+2(\varepsilon_1+\varepsilon_2+\varepsilon_3)F_1^TF_1+\varepsilon_1^{-1}M_1M_1^T,$$
$$\Theta_{12}=M_1B(t)+A^T(t)M_2^T+N_1-N_2^T,\ \Theta_{13}=P+A^T(t)M_3^T-M_1-N_3^T,$$
$$\Theta_{22}=M_2B(t)+B^T(t)M_2^T-(1-\tau)Q+N_2+N_2^T+2(\varepsilon_1+\varepsilon_2+\varepsilon_3)F_2^TF_2+\varepsilon_2^{-1}M_2M_2^T,$$
$$\Theta_{23}=B^T(t)M_3^T-M_2+N_3^T,\ \Theta_{33}=-M_3-M_3^T+h^2R+\varepsilon_3^{-1}M_3M_3^T$$

If $\Theta<0$, then $L\tilde{V}(t)<0$.

By Schur complement, inequality $\Theta<0$ is equivalent to

$$\Theta_1<0 \tag{18}$$

where

$$\Theta_1=\begin{bmatrix}\Theta_{11} & \Theta_{12} & \Theta_{13} & N_1 & C^T(t)P\\ * & \Theta_{22} & \Theta_{23} & N_2 & D^T(t)P\\ * & * & \Theta_{33} & N_3 & 0\\ * & * & * & -R & 0\\ * & * & * & * & -P\end{bmatrix}$$

Set

$$\Pi=\begin{bmatrix}\Pi_{11} & \Pi_{12} & \Pi_{13} & N_1 & C^TP\\ * & \Pi_{22} & \Pi_{23} & N_2 & D^TP\\ * & * & \Pi_{33} & N_3 & 0\\ * & * & * & -R & 0\\ * & * & * & * & -P\end{bmatrix}$$

with

$$\Pi_{11}=M_1A+A^TM_1^T+Q-N_1-N_1^T+2(\varepsilon_1+\varepsilon_2+\varepsilon_3)F_1^TF_1+\varepsilon_1^{-1}M_1M_1^T,$$
$$\Pi_{12}=M_1B+A^TM_2^T+N_1-N_2^T,\ \Pi_{13}=P+A^TM_3^T-M_1-N_3^T,$$
$$\Pi_{22}=M_2B+B^TM_2^T-(1-\tau)Q+N_2+N_2^T+2(\varepsilon_1+\varepsilon_2+\varepsilon_3)F_2^TF_2+\varepsilon_2^{-1}M_2M_2^T,$$
$$\Pi_{23}=B^TM_3^T-M_2+N_3^T,\ \Pi_{33}=-M_3-M_3^T+h^2R+\varepsilon_3^{-1}M_3M_3^T$$

Then, through condition (18), we have

$$\Theta_1=\Pi+\Sigma_1F(t)\Sigma_2+\Sigma_2^TF^T(t)\Sigma_1^T+\Sigma_3F(t)\Sigma_4+\Sigma_4^TF^T(t)\Sigma_3^T<0 \tag{19}$$

where

$$\Sigma_1 = [H^T M_1^T \;\; H^T M_2^T \;\; H^T M_3^T \;\; 0 \;\; 0]^T,\ \Sigma_2 = [E_1 \;\; E_2 \; 0 \;\; 0 \;\; 0],$$
$$\Sigma_3 = [0 \;\; 0 \;\; 0 \;\; 0 \;\; H^T P]^T,\ \Sigma_4 = [E_3 \;\; E_4 \; 0 \;\; 0 \;\; 0] \tag{20}$$

By Lemma 2, (19) holds if

$$\Pi + \mu_1 \Sigma_2^T \Sigma_2 + \mu_1^{-1} \Sigma_1 \Sigma_1^T + \mu_2 \Sigma_4^T \Sigma_4 + \mu_2^{-1} \Sigma_3 \Sigma_3^T < 0 \tag{21}$$

Then, by Schur complement and substituting of (20) into (21), we see

$$\begin{bmatrix} \Pi_{11} + \mu_1 E_1^T E_1 + \mu_2 E_3^T E_3 & \Pi_{12} + \mu_1 E_1^T E_2 + \mu_2 E_3^T E_4 & \Pi_{13} & N_1 & C^T P & M_1 H & 0 \\ * & \Pi_{22} + \mu_1 E_2^T E_2 + \mu_2 E_4^T E_4 & \Pi_{23} & N_2 & D^T P & M_2 H & 0 \\ * & * & \Pi_{33} & N_3 & 0 & M_3 H & 0 \\ * & * & * & -R & 0 & 0 & 0 \\ * & * & * & * & -P & 0 & PH \\ * & * & * & * & * & -\mu_1 I & 0 \\ * & * & * & * & * & * & -\mu_2 I \end{bmatrix} < 0 \tag{22}$$

By Schur complement, (22) is equivalent to (4), which implies that systems (1) is of stochastically asymptotical stability. This completes the proof.

4 Conclusions

The delay dependent stability problem has been investigated for the stochastic system with nonlinearity, uncertainty and time-varying delays. The new stability criterion has been derived by introducing some free-weighting matrices, which has more freedom to determine the stability of the systems. The proposed result is formulated in terms of linear matrix inequality, which can be efficiently solved by standard convex optimization algorithms.

Acknowledgements

This work is supported by the Natural Science Foundation of Hubei Province, China(No.2008CDB069), the Major Research Program of Hubei Provincial Department of Education, China(No.Z20092701), the Innovative Group Project of Hubei Provincial Department of Education, China(No. 03BA85), and the Excellent Youth Project of Hubei Provincial Department of Education, China(No. Q20082703).

References

1. Xu, S., Chen, T.: Robust H□∞□ control for uncertain stochastic systems with state delay. IEEE Transactions on Automatic Control 47(12), 2089–2094 (2002)
2. Liao, X., Mao, X.: Exponential stability of stochastic delay interval systems. Systems & Control Letters 40, 171–181 (2000)

3. Xu, S., Shi, P., Chu, Y., Zou, Y.: Robust stochastic stabilization and H_∞ control of uncertain neutral stochastic time-delay systems. Journal of Mathematical Analysis and Applications 314, 1–6 (2006)
4. Huang, L., Deng, F.: Robust exponential stabilization of stochastic large-scale delay systems. In: Proceedings of 2007 IEEE international conference on control and automation, pp. 107–112 (2007)
5. Yue, D., Han, Q.L.: Delay-dependent exponential stability of stochastic systems with time-varying, nonlinearity and Markovian switching. IEEE Transactions on Automatic Control 50, 217–222 (2005)
6. Lu, H.Q., Zhou, W.N.: Delay-dependent robust H_∞ control for uncertain stochastic systems. Control and Decision 24(1), 76–80 (2009)
7. Huang, L.R., Mao, X.: Robust delayed-state-feedback stabilization of uncertain stochastic systems. Automatic 45, 1332–1339 (2009)
8. Yue, D., Fang, J., Won, S.: Delay-dependent robust stability of stochastic uncertain systems with time delay and markovian jump parameters. Circuits Syst. Signal Processing 22, 351–365 (2003)
9. Chen, W.H., Guan, Z.H., Lu, X.M.: Delay-dependent exponential stability of uncertain stochastic systems with multiple delays: an LMI approach. Systems & Control Letters 54, 547–555 (2005)
10. Wei, G.L., Wang, Z.D., Shu, H.S.: Nonlinear H_∞ control of stochastic time-delay systems with Markovian switching. Chao, Solitons and Fractals 35, 442–451 (2008)
11. Gao, H.J., Lam, J., Wang, C.H.: Robust energy-to-peak filter design for stochastic time-delay systems. Systems & Control Letters 55, 101–111 (2006)
12. Xu, S.Y., Lam, J.: Exponential H_∞ filter design for uncertain Takagi-Sugeno fuzzy systems with time delay. Artificial Intelligence 17, 645–659 (2004)
13. Chen, Y., Xue, A.K., Wang, J.H.: Delay-dependent passive control of stochastic delay systems. Acta Automatica Sinica 35(3), 324–327 (2009)

Stability Analysis of Fuzzy Cohen-Grossberg Neural Networks with Distributed Delays and Reaction-Diffusion Terms

Weifan Zheng[1,2] and Jiye Zhang[1,*]

[1] National Traction Power Laboratory, Southwest Jiaotong University(SWJTU)
[2] Information Research Institute, SWJTU, Chengdu 610031, People's Republic of China
jyzhang@home.swjtu.edu.cn, wfzheng@home.swjtu.edu.cn

Abstract. In this paper, we investigate a class of fuzzy Cohen-Grossberg neural networks (FCGNN) with distributed delays and reaction-diffusion terms. Based on the properties of M-matrix, by constructing vector Liapunov functions and applying differential inequalities, the sufficient conditions ensuring existence, uniqueness, and global exponential stability of the equilibrium point of FCGNN with distributed delays and reaction-diffusion terms are obtained.

Keywords: Stability, FCGNN, Reaction-diffusion, Distributed Delays.

1 Introduction

In recent years, Cohen-Grossberg neural network (CGNN) [1], have attracted the attention of the scientific community due to their promising potential for tasks of associative memory, parallel computation and their ability to solve difficult optimization problems. In these applications it is required that there is a well-defined computable solution for all possible initial states. This means that the neural network should have a unique equilibrium point that is globally stable. Thus, the qualitative analysis of dynamic behaviors is a prerequisite step for the practical design and application of neural networks. There are many papers discuss the qualitative properties for neural networks [2], and fuzzy cellular neural network (FCNN) [3]. In hardware implementation, time delays are unavoidable, and may lead to an oscillation and instability of networks [4]. In most situations, the time delays are variable, and in fact unbounded. Therefore, the study of stability of neural networks with variable and unbounded delay is practically important. Stability of neural networks and CGNN with constant and variable time delays stand by differential equation has been studied in [4~6]. The global exponential stability of CGNN with reaction-diffusion terms and distributed delays are studied in [7~8].

In this paper, we study FCGNN with reaction-diffusion terms, which contain both variable time delays and unbounded delay. We relax some conditions on activation functions and diffusion functions of systems similar to that discussed in [6~9], by

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 684–692, 2010.

using M-matrix theory and nonlinear integro-differential inequalities, even type Liapunov functions were constructed to analyze the conditions ensuring the existence, uniqueness and global exponential stability of the equilibrium point of the models.

2 Model Description and Preliminaries

In this paper, we analyze the stability of reaction-diffusion FCGNN with both variable delays and unbounded delay described by the following differential equations

$$\frac{\partial u_i(t)}{\partial t} = \sum_{k=1}^{m} \frac{\partial}{\partial x_k}[D_{ik}(t,x,u_i)\frac{\partial u_i}{\partial x_k}] - d_i(u_i(t))[\rho_i(u_i(t)) - \sum_{j=1}^{n} a_{ij} f_j(u_j(t)) - \sum_{j=1}^{n} b_{ij} g_j(u_j(t-\tau_{ij}(t)))$$

$$- \sum_{j=1}^{n} b_{ij} g_j(u_j(t-\tau_{ij}(t))) - \wedge_{j=1}^{n} c_{ij}^1 \int_{-\infty}^{t} k_{ij}(t-s) h_j(u_j(s)) ds$$

$$- \vee_{j=1}^{n} c_{ij}^2 \int_{-\infty}^{t} k_{ij}(t-s) h_j(u_j(s)) ds + J_i] \tag{1}$$

$$\frac{\partial u_i}{\partial \tilde{n}} = col(\frac{\partial u_i}{\partial x_1}, ..., \frac{\partial u_i}{\partial x_m}) = 0 \quad t \in I,\ x \in \partial\Omega, \quad i=1,2,\ldots,n \tag{2}$$

where u_i is the state of neuron i, $(i = 1,2,\cdots,n)$ and n is the number of neurons; $D_i(t,x,u_i)$ is smooth reaction-diffusion function, $A = (a_{ij})_{n\times n}$, $B = (b_{ij})_{n\times n}$ are connection matrices, $J = (J_1,\cdots,J_n)^{\mathrm{T}}$ is the constant input vector. $f(u) = (f_1(u_1),...,f_n(u_n))^{\mathrm{T}}$, $g(u) = (g_1(u_1),\cdots,g_n(u_n))^{\mathrm{T}}$, and $h(u) = (h_1(u_1),...,h_n(u_n))^{\mathrm{T}}$ are the activation functions of the neurons; $d_i(u_i)$ represents an amplification function; $\rho_i(u_i)$ is an appropriately behaved function such that the solutions of model (1) remain bounded. The variable delays $\tau_{ij}(t)$ ($i,j = 1,2,\cdots,n$) are bounded functions, i.e. $0 \le \tau_{ij}(t) \le \tau$, and $k_{ij}:[0,\infty) \to [0,\infty)$, $(i,j = 1,2,\cdots,n)$ are piecewise continuous on $[0,\infty)$. Let $k = (k_{ij})_{n\times n}$. $\wedge$ and $\vee$ denote the fuzzy AND and fuzzy OR operation, respectively. $C^1 = (c_{ij}^1)_{n\times n}$ and $C^2 = (c_{ij}^2)_{n\times n}$ are elements matrices of fuzzy feedback MIN template, fuzzy feedback MAX template, respectively.

The conventional conditions for kernel functions of (1) meet the following assumptions:

Assumption A: $\int_0^\infty k_{ij}(s)ds = 1$, $\int_0^\infty s k_{ij}(s)ds < +\infty$, $(i,j = 1,2,\cdots,n)$.

Assumption B: $\int_0^\infty k_{ij}(s)ds = 1$, $\int_0^\infty e^{\beta s} k_{ij}(s)ds = K_{ij} < +\infty$, $(i,j = 1,2,\cdots,n)$.

In order to study the exponential stability of neural networks (1) conveniently, we inquire kernel functions meet the following assumption:

Assumption C: $\int_0^\infty e^{\beta s} k_{ij}(s)ds = N_{ij}(\beta)$, $(i,j = 1,2,\cdots,n)$, where $N_{ij}(\beta)$ are continuous functions in $[0,\delta)$, $\delta > 0$, and $N_{ij}(0) = 1$.

It is easy to prove that the Assumption C includes Assumption A and B [5].

The initial conditions of (1) are of the form $u_i(s) = \phi_i(s)$, $s \le 0$, where ϕ_i is bounded and continuous on $(-\infty,0]$. (2) is the boundary condition of (1), in which $x \in \Omega \subset R^m$, Ω is a compact set with smooth boundary and $mes\Omega > 0$, $\partial\Omega$ is the boundary of Ω , $t \in I = [0,+\infty]$.

For convenience, we introduce some notations. The express $u = (u_1,u_2,...,u_n)^T \in R^n$ represents a column vector (the symbol $(\,)^T$ denotes transpose). For matrix $A = (a_{ij})_{n\times n}$, $|A|$ denotes absolute value matrix given by $|A| = (|a_{ij}|)_{n\times n}$, $i, j = 1, 2,\ldots, n$; $[A]^S$ is defined as $(A^{\mathrm{T}} + A)/2$. For $x \in R^n$, $|x| = (|x_1|,...,|x_n|)^{\mathrm{T}}$, $\|x\|$ denotes a vector norm defined by $\|x\| = \max\limits_{1\le i\le n}\{|x_i|\}$. $\overline{\rho} = \mathrm{diag}(\rho_1,...,\rho_n)$, $\overline{d} = \mathrm{diag}(d_1,...,d_n)$. And

$$D_i(t,x,u_i) = \sum_{k=1}^{m} \frac{\partial}{\partial x_k}(D_{ij}(t,x,u_i)\frac{\partial u_i}{\partial x_k}),$$

So model (1) becomes the following system:

$$\dot{u}_i(t) = D_i(t,x,u_i(t)) - d_i(u_i(t))[\rho_i(u_i(t)) - \sum_{j=1}^{n} a_{ij} f_j(u_i(t)) - \sum_{j=1}^{n} b_{ij} g_j(u_i(t-\tau_{ij}(t))$$

$$- \wedge_{j=1}^{n} c_{ij}^1 \int_{-\infty}^{t} k_{ij}(t-s) h_j(u_j(s)) ds \;\; - \vee_{j=1}^{n} c_{ij}^2 \int_{-\infty}^{t} k_{ij}(t-s) h_j(u_j(s)) ds + J_i] . \qquad (3)$$

Therefore, system (3) and (1) has the same properties of stability.

Now we consider the activation functions of the neurons, amplification function and behaved function satisfying the following assumption:

Assumption D: For each $i \in \{1,2,\cdots,n\}$, $f_i : \mathrm{R} \to \mathrm{R}$, $g_i : \mathrm{R} \to \mathrm{R}$ and $h_i : \mathrm{R} \to \mathrm{R}$, there exist the real numbers $p_i > 0$, $q_i > 0$ and $r_i > 0$ such that

$$p_i = \sup_{y\ne z} |\frac{f_i(y) - f_i(z)}{y - z}|, \; q_i = \sup_{y\ne z} |\frac{g_i(y) - g_i(z)}{y - z}|, \; r_i = \sup_{y\ne z} |\frac{h_i(y) - h_i(z)}{y - z}|$$

for every $y \ne z$. Let $P = \mathrm{diag}(p_1,...,p_n)$, $Q = \mathrm{diag}(q_1,...,q_n)$, $R = \mathrm{diag}(r_1,...,r_n)$.

Assumption D introduced the supremum of Global/Local Lipschitz constants, and expanded the scope of system application. So, the activation functions such as sigmoid type and piecewise linear type are the special case of that satisfying it.

Assumption E: For each $i \in \{1,2,\cdots,n\}$, $e_i : \mathrm{R} \to \mathrm{R}$ is strictly monotone increasing, i.e. there exists a positive diagonal matrix $\overline{\rho} = \mathrm{diag}(\rho_1,\rho_2,\cdots,\rho_n) > 0$ such that

$$\frac{\rho_i(u) - \rho_i(v)}{u - v} \ge \rho_i, (u \ne v) \;\; .$$

Assumption F: For each $i \in \{1,2,\cdots,n\}$, $d_i : \mathrm{R} \to \mathrm{R}$ is continuous function and $0 < \sigma_i \le d_i$, where σ_i is a constant.

3 Existence and Uniqueness of the Equilibrium Point

In this section, we shall study the condition which ensures the existence and uniqueness of the equilibrium point of system (1).

For convenience, we introduce some definitions and lemmas as follows.

Definition 1 [10]. A real matrix $A=(a_{ij})_{n\times n}$ is said to be an M-matrix if $a_{ij}\le 0$ $i,j=1,2,...,n$, $i\ne j$, $a_{ii}>0$ and all successive principal minors of A are positive.

Definition 2. The equilibrium point u^* of (1) is said to be globally exponentially stable, if there exist constant $\lambda>0$ and $\beta>0$ such that $\|u(t)-u^*\|\le\beta\|\phi-u^*\|e^{-\lambda t}$ ($t\ge 0$), where $\|\phi-u^*\|=\max\limits_{1\le i\le n}\sup\limits_{s\in[-\tau,0]}|\phi_i(s)-u_i^*|$.

If there is a constant $u_0=u_0^*=const$ (const denotes invariable constant) which is the solution of the following equations:

$$-\rho_i(u_i(t))+\sum_{j=1}^{n}a_{ij}f_j(u_i(t))+\sum_{j=1}^{n}b_{ij}g_j(u_i(t-\tau_{ij}(t))$$
$$-\wedge_{j=1}^{n}c_{ij}^1\int_{-\infty}^{t}k_{ij}(t-s)h_j(u_j(s))ds-\vee_{j=1}^{n}c_{ij}^2\int_{-\infty}^{t}k_{ij}(t-s)h_j(u_j(s))ds+J_i=0 \qquad (4)$$

then $\frac{\partial u_i^*}{\partial x}=0$. That is to say (4) and (3) have the same equilibrium point, and so, system (4) has the same equilibrium point as that of system (1).

We firstly study the solutions of the nonlinear map associated with (1) as follows:

$$H_i(u_i)=-\rho_i(u_i)+\sum_{j=1}^{n}a_{ij}f_j(u_i)+\sum_{j=1}^{n}b_{ij}g_j(u_i)+\wedge_{j=1}^{n}c_{ij}^1h_j(u_j)+\vee_{j=1}^{n}c_{ij}^2h_j(u_j)+J_i \qquad (5)$$

Let $H(x)=(H_1(u_1),H_2(u_2),...,H_n(u_n))^T$. It is well known that the solutions of $H(u)=0$ are equilibriums in (1). If map $H(u)=0$ is a homeomorphism on R^n, then system (1) has a unique equilibrium u^* (see [2]). In the following, we will give condition ensuring $H(u)=0$ is a homeomorphism.

Lemma 1 [5]. If $H(u)\in C^0$ satisfies the following conditions: (i) $H(u)$ is injective on R^n; (ii) $\|H(u)\|\to\infty$ as $\|u\|\to\infty$; then $H(u)$ is a homeomorphism of R^n.

Lemma 2 [3]. Suppose x and y are two states of system (1), then

$$|\wedge_{j=1}^{n}c_{ij}^1h_j(x_j)-\wedge_{j=1}^{n}c_{ij}^1h_j(y_j)|\le\sum_{j=1}^{n}|c_{ij}^1||h_j(x_j)-h_j(y_j)|,$$
$$|\vee_{j=1}^{n}c_{ij}^2h_j(x_j)-\vee_{j=1}^{n}c_{ij}^2h_j(y_j)|\le\sum_{j=1}^{n}|c_{ij}^2||h_j(x_j)-h_j(y_j)|.$$

Theorem 1. If Assumption D, E, and F are satisfied, and $\alpha = \overline{\rho} - (|A|P + |B|Q + |C^1|R + |C^2|R)$ is an M-matrix, then, for every input J, system (1) has a unique equilibrium u^*.

Proof: In order to prove that for every input J, system (1) has a unique equilibrium point u^*, it is only to prove that $H(u)$ is a homeomorphism on R^n. In following, we shall prove it in two steps.

Step 1, we will prove that condition (i) in Lemma 1 is satisfied. Suppose, for purposes of contradiction, that there exist $x, y \in \mathrm{R}^n$ with $x \neq y$ such that $H(x) = H(y)$.

From Assumption E, we know that there exists matrix $\overline{\beta} = \mathrm{diag}(\beta_1, \beta_2, \cdots, \beta_n)$, $(\beta_i \geq \rho_i)$ such that $\rho_i(x_i) - \rho_i(y_i) = \beta_i(x_i - y_i)$, for $i = 1,2,...,n$.

Form (5), we get

$$-[\rho_i(x_i) - \rho_i(y_i)] + \sum_{j=1}^{n} a_{ij}(f_j(x_j) - f_j(y_j)) + \sum_{j=1}^{n} b_{ij}(g_j(x_j) - g_j(y_j))$$
$$+ \wedge_{j=1}^{n} c_{ij}^1 h_j(x_j) - \wedge_{j=1}^{n} c_{ij}^1 h_j(y_j) + \vee_{j=1}^{n} c_{ij}^2 h_j(x_j) - \vee_{j=1}^{n} c_{ij}^2 h_j(y_j)) = 0. \quad (6)$$

We have $|\rho_i(x_i) - \rho_i(y_i)| \leq |\sum_{j=1}^{n} a_{ij}(f_j(x_j) - f_j(y_j))| + |\sum_{j=1}^{n} b_{ij}(g_j(x_j) - g_j(y_j))|$

$+ |\wedge_{j=1}^{n} c_{ij}^1 h_j(x_j) - \wedge_{j=1}^{n} c_{ij}^1 h_j(y_j)| + |\vee_{j=1}^{n} c_{ij}^2 h_j(x_j) - \vee_{j=1}^{n} c_{ij}^2 h_j(y_j))|$, $(i = 1,2,...,n)$

From Assumption D, E and Lemma 2, we get

$$[\overline{\beta} - (|A|P + |B|Q + |C^1|R + |C^2|R)]|x - y| \leq 0. \quad (7)$$

Because of α being an M-matrix, from Lemma 1, we know that all elements of $(\overline{\beta} - (|A|P + |B|Q + |C^1|R + |C^2|R))^{-1}$ are nonnegative. Therefore $|x - y| \leq 0$, i.e., $x = y$. From the supposition $x \neq y$, thus this is a contradiction. So $H(u)$ is injective.

Step 2. We now prove that condition (ii) in Lemma 1 is satisfied. Let $\overline{H}(u) = H(u) - H(0))$. From (5), we get

$$\overline{H}_i(u_i) = -\beta_i u_i + \sum_{j=1}^{n} a_{ij}(f_j(u_i) - f_j(0)) + \sum_{j=1}^{n} b_{ij}(g_j(u_i) - g_j(0))$$
$$+ \wedge_{j=1}^{n} c_{ij}^1 h_j(u_j) - \wedge_{j=1}^{n} c_{ij}^1 h_j(0) + \vee_{j=1}^{n} c_{ij}^2 h_j(u_j) - \vee_{j=1}^{n} c_{ij}^2 h_j(0)), \ (i = 1, 2, \ldots, n).$$

To show that $H(u)$ is homeomorphism, it suffices to show that $\overline{H}(u)$ is homeomorphism. According to Assumption D, we get

$$|f_i(u) - f_i(0)| \leq p_i |u|, |g_i(u) - g_i(0)| \leq q_i |u|, |h_i(u) - h_i(0)| \leq r_i |u|, (i = 1, 2, \ldots, n).$$

Since $\alpha = \overline{\rho} - (|A|P + |B|Q + |C^1|R + |C^2|R)$ is an M-matrix, so $\overline{\alpha} = \overline{\beta} - (|A|P + |B|Q + |C^1|R + |C^2|R)$ is an M-matrix. From the property of M-matrix [10], there exists a matrix $T = \mathrm{diag}(T_1, \cdots, T_n) > 0$ such that

$$[D(-\overline{\beta}+|A|P+|B|Q+|C^1|R+|C^2|R)]^S \le -\varepsilon E_n < 0 \tag{8}$$

for sufficiently small $\varepsilon > 0$, where E_n is the identity matrix. Calculating

$$[Tu]^{\mathrm{T}} \overline{H}(u) = \sum_{i=1}^{n} u_i T_i [-d_i u_i + \sum_{j=1}^{n} a_{ij}(f_j(u_i) - f_j(0)) + \sum_{j=1}^{n} b_{ij}(g_j(u_i) - g_j(0))$$

$$+ \sum_{i=1}^{n} (\wedge_{j=1}^{n} c_{ij}^1 h_j(u_j) - \wedge_{j=1}^{n} c_{ij}^1 h_j(0)) + \sum_{i=1}^{n} (\vee_{j=1}^{n} c_{ij}^2 h_j(u_j) - \vee_{j=1}^{n} c_{ij}^2 h_j(0))]$$

$$\le |u|^{\mathrm{T}} [T(-\overline{\beta}+|A|P+|B|Q+|C^1|R+|C^2|R)]^S |u| \le -\varepsilon \|u\|^2. \tag{9}$$

From (9) and using Schwartz inequality, we get $\varepsilon \|u\|^2 \le \|T\| \|u\| \|\overline{H}(u)\|$, namely,

$$\varepsilon \|u\| \le \|T\| \|\overline{H}(u)\| . \tag{10}$$

So, $\|\overline{H}(u)\| \to +\infty$, i.e., $\|H(u)\| \to +\infty$ as $\|u\| \to +\infty$. From Lemma 1, we know that for every input J, map $H(u)$ is homeomorphism on R^n. So systems (1) have a unique equilibrium point u^*. The proof is completed.

4 Global Exponential Stability of Equilibrium Point

In this section, we shall apply the ideal of vector Lyapunov method to analyze global exponential stability of model (1).

Theorem 2. If Assumption D, E and F are satisfied and $\alpha = \overline{\rho} - (|A|P+|B|Q+|C^1|R+|C^2|R)$ is an M-matrix, then for each input J, systems (1) have a unique equilibrium point, which is globally exponentially stable.

Proof: Since α is an M-matrix, from Theorem 1, system (1) has a unique equilibrium point u^*. Let $z(t) = u(t) - u^*$, model (1) can be written as

$$\frac{\partial z_i(t)}{\partial t} = \sum_{k=1}^{m} \frac{\partial}{\partial x_k} [D_{ik}(t, x, (z_i(t) + u^*)) \frac{\partial z_i}{\partial x_k}] - d_i(z_i(t))[\rho_i(z_i(t)) - \sum_{j=1}^{n} a_{ij} f_j(z_j(t))$$

$$- \sum_{j=1}^{n} b_{ij} g_j(z_j(t - \tau_{ij}(t))) - \wedge_{j=1}^{n} c_{ij}^1 \int_{-\infty}^{t} k_{ij}(t-s) h_j(z_j(s) + u_j^*) ds + \wedge_{j=1}^{n} c_{ij}^1 h_j(u_j^*)$$

$$- \vee_{j=1}^{n} c_{ij}^2 \int_{-\infty}^{t} k_{ij}(t-s) h_j(z_j(s) + u_j^*) ds + \vee_{j=1}^{n} c_{ij}^2 h_j(u_j^*)] \ (i = 1,2,...,n), \tag{11}$$

where $f_j(z_j(t)) = f_j(z_j(t) + u_j^*) - f_j(u_j^*)$, $h_j(z_j(t)) = h_j(z_j(t) + u_j^*) - h_j(u_j^*)$,

$$g_j(z_j(t - \tau_{ij}(t))) = g_j(z_j(t - \tau_{ij}(t) + u_j^*)) - g_j(u_j^*)$$

$$d_i(z_i(t)) = d_i(z_i(t) + u_i^*), \ \rho_i(z_i(t)) = \rho_i(z_i(t) + u_i^*) - \rho_i(u_i^*).$$

The initial condition of (11) is $\psi(s)=\phi(s)-u^*$, $s\le 0$, and (11) have a unique equilibrium at $z=0$. So according to the Assumption D, we get

$$|f_j(z_j(t))|\le p_j|z_j(t)|,|g_j(z_j(t-\tau_{ij}(t)))|\le p_j|z_j(t-\tau_{ij}(t))|,|h_j(z_j(t))|\le r_j|z_j(t)|.$$

Due to α being an M-matrix, so $\overline{\alpha}=\overline{\beta}-(|A|P+|B|Q+|C^1|R+|C^2|R)$ is an M-matrix. Using property of M-matrix [10], there exist $\xi_i>0 \quad (i=1,2,...,n)$ satisfy

$$-\xi_i\beta_i+\sum_{j=1}^{n}\xi_j(|a_{ij}|p_j+|b_{ij}|q_j+|c_{ij}^1|r_j+|c_{ij}^2|r_j)<0 \quad (i=1,2,...,n). \tag{12}$$

From Assumption F, we know that $0\le\sigma_i\le d_i(z_i(t)+u_i^*)$, so $d_i(z_i(t)+u_i^*)/\sigma_i\ge 1$.

So there exist a constant $\lambda>0$ such that

$$-\xi_i(\beta_i-\frac{\lambda}{\sigma_i})+\sum_{j=1}^{n}\xi_j(|a_{ij}|p_j+e^{\lambda\tau}|b_{ij}|q_j+(|c_{ij}^1|+|c_{ij}^2|)N_{ij}(\lambda)r_j)<0. \tag{13}$$

Here, τ is a fixed number according to assumption of neural networks (1). Let $V_i(t)=e^{\lambda t}z_i(t) \quad (i=1,2,...,n)$, and Lyapunov function $\overline{V_i(t)}=\int_\Omega|V_i(t)|dx$, calculating the upper right derivative $D^+\overline{V_i(t)}$ of $\overline{V_i(t)}$ along the solutions of (11), we get

$$D^+\overline{V_i(t)}=\int_\Omega D^+|V_i(t)|dx=\int_\Omega(e^{\lambda t}\operatorname{sgn}z_i\dot{z}_i+\lambda e^{\lambda t}|z_i|)dx. \tag{14}$$

From (11) and (14), according to Assumption D, boundary condition (2), and $mes\Omega>0$, we get

$$\begin{aligned}
D^+\overline{V_i(t)}=&\int_\Omega e^{\lambda t}\operatorname{sgn}z_i(t)\sum_{k=1}^{m}\frac{\partial}{\partial x_k}[D_{ik}(t,x,(z_i(t)+u^*))\frac{\partial u_i}{\partial x_k}]dx+\int_\Omega\lambda e^{\lambda t}|z_i(t)|dx\\
&-\int_\Omega d_i(z_i(t))e^{\lambda t}\{\operatorname{sgn}z_i(t)[\rho_i(z_i(t))-\sum_{j=1}^{n}a_{ij}f_j(z_j(t))-\sum_{j=1}^{n}b_{ij}g_j(z_j(t-\tau_{ij}(t)))]\}dx\\
&+\int_\Omega d_i(z_i(t))e^{\lambda t}\operatorname{sgn}z_i(t)\times[\wedge_{j=1}^{n}c_{ij}^1\int_{-\infty}^{t}k_{ij}(t-s)h_j(z_j(s)+u_j^*)ds-\wedge_{j=1}^{n}c_{ij}^1h_j(u_j^*)\\
&+\vee_{j=1}^{n}c_{ij}^2\int_{-\infty}^{t}k_{ij}(t-s)h_j(z_j(s)+u_j^*)ds-\vee_{j=1}^{n}c_{ij}^2h_j(u_j^*)]dx\\
\le&-\int_\Omega d_i(z_i(t))[(\beta_i-\frac{\lambda}{\sigma_i})|V_i(t)|-\sum_{j=1}^{n}|a_{ij}|p_j|V_j(t)|-e^{\lambda\tau_{ij}(t)}|b_{ij}|q_j|V_j(t-\tau_{ij}(t))|]dx\\
&+\int_\Omega d_i(z_i(t))[\sum_{j=1}^{n}(|c_{ij}^1|+|c_{ij}^2|)r_j\times\int_{-\infty}^{t}k_{ij}(t-s)e^{\lambda(t-s)}|V_j(s)|ds]dx
\end{aligned} \tag{15}$$

Defining the curve $\gamma=\{y(l):y_i=\xi_i l,l>0,i=1,2,...,n\}$, and the set $\Omega(y)=\{u:0\le u\le y,y\in\gamma\}$. Let $\xi_{\min}=\min_{1\le i\le n}\{\xi_i\}$, $\xi_{\max}=\max_{1\le i\le n}\{\xi_i\}$. Taking $l_0=(1+\delta)\|\psi\|/\xi_{\min}$, where $\delta>0$ is a constant number, then $\{|V|:|V|=e^{\lambda s}|\psi(s)|, \quad -\tau\le s\le 0\}\subset\Omega(z_0(l_0))$, namely $|V_i(s)|=e^{\lambda s}|\psi_i(s)|<\xi_i l_0$, $-\tau\le s\le 0$, $i=1,2,\ldots,n$.

We claim that $|V_i(t)| < \xi_i l_0$, for $t \in [0,+\infty)$, i=1, 2, … , n . If it is not true, then there exist some index i and t_1 ($t_1 > 0$) such that $|V_i(t_1)| = \xi_i l_0$, $D^+|V_i(t_1)| \geq 0$, and $|V_j(t)| \leq \xi_j l_0$, for $-\tau < t \leq t_1$, j =1, 2, … , n. So we could get $D^+\overline{V_i(t_1)} = \int_\Omega D^+|V_i(t_1)|dx \geq 0$. However, from (13) and (15), we get

$$D^+\overline{V_i(t_1)} = \int_\Omega D^+|V_i(t_1)|dx \leq -d_i(z_i(t))[\xi_i(\beta_i - \frac{\lambda}{\sigma_i}) - \sum_{j=1}^{n}\xi_j(|a_{ij}|p_j$$

$$+ e^{\lambda\tau}|b_{ij}|q_j + (|c_{ij}^1| + |c_{ij}^2|)N_{ij}(\lambda)r_j)]l_0 < 0$$

There is a contradiction. So $|V_i(t)| < \xi_i l_0$, for $t \in [0,+\infty)$, therefore,

$$|z_i(t)| < \xi_i l_0 e^{-\lambda t} \leq (1+\delta)\|\psi\|\xi_{\max}/\xi_{\min}\ e^{-\lambda t} = \beta\|\psi\|\ e^{-\lambda t},$$

where $\beta = (1+\delta)\xi_{\max}/\xi_{\min}$. From definition 2, the zero solution of systems (11) is globally exponentially stable, i.e., the equilibrium point of systems (1) is globally exponentially stable. The proof is completed.

It is obvious that the result in theorem 2 extended that in [6~9].

5 Conclusions

In the paper, a thorough analysis of existence, uniqueness, and global exponential stability of the equilibrium point for FCGNN with reaction-diffusion terms and both variable delays and unbounded delay have been presented. The conditions ensuring the existence and uniqueness of the equilibrium are obtained. By constructing proper Liapunov functionals, using M-matrix theory and qualitative property of the differential inequalities, the sufficient conditions for global exponential stability of the equilibrium point for FCGNN are obtained. Some restrictions on FCGNN are removed, and the results in this paper are explicit and convenient to verify in practice.

Acknowledgments. This work is supported by Natural Science Foundation of China (No. 10772152, 60974132), Doctoral Fund of Ministry of Education of China (No. 200806130003).

References

[1] Cohen, M.A., Grossberg, S.: Absolute Stability and Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks. IEEE Trans. Syst. Man Cybern. 13(5), 815–825 (1983)

[2] Forti, M., Tesi, A.: New Conditions for Global Stability of Neural Networks with Application to Linear and Quadratic Programming Problems. IEEE Trans. Circuits System-I 42(7), 354–366 (1995)

[3] Yang, T., Yang, L.B.: Exponential Stability of Fuzzy Cellular Neural Networks with Constant and Time-Varying Delays. IEEE Trans. Circ. Syst.-I 43, 880–883 (1996)

[4] Civalleri, P.P., Gill, L.M., Pandolfi, L.: On Stability of Cellular Neural Networks with Delay. IEEE Transactions on Circuits and Systems-I 40(3), 157–164 (1993)
[5] Zhang, J., Suda, Y., Iwasa, T.: Absolutely Exponential Stability of a Class of Neural Networks with Unbounded Delay. Neural Networks 17, 391–397 (2004)
[6] Zhang, J., Suda, Y., Komine, H.: Global Exponential Stability of Cohen-Grossberg Neural Networks with Variable Delays. Physics Letters A 338, 44–50 (2005)
[7] Zheng, W., Zhang, J.: Exponential Stability of Cohen-Grossberg Neural Networks with Reaction- diffusion Terms and Both Variable Time Delays and Unbounded Delay. DCDIS A Supplement, Advances in Neural Networks 14(S1), 100–105 (2007)
[8] Song, Q., Cao, J.: Global Exponential Robust Stability of Cohen-Grossberg Neural Network with Time-varying Delays and Reaction-diffusion Terms. Journal of the Franklin Institute 343(7), 705–719 (2006)
[9] Zhang, J., Ren, D., Zhang, W.: Global Exponential Stability of Fuzzy Cohen-Grossberg Neural Networks with Variable Delays and Distributed Delays. In: Huang, D.-S., Heutte, L., Loog, M. (eds.) ICIC 2007. LNCS (LNAI), vol. 4682, pp. 66–74. Springer, Heidelberg (2007)
[10] Siljak, D.D.: Large-scale Dynamic Systems — Stability and Structure. Elsevier North-Holland, Inc., New York (1978)

Global Exponential Robust Stability of Delayed Hopfield Neural Networks with Reaction-Diffusion Terms

Xiaohui Xu, Jiye Zhang, and Weihua Zhang

Traction Power State Key Laboratory, Southwest Jiaotong University,
Chengdu 610031, China
jyzhang@home.swjtu.edu.cn

Abstract. Some conditions ensuring existence, uniqueness for a class of delayed interval Hopfield neural networks with reaction-diffusion terms are proposed in this paper. Applying vector Lyapunov function and M-matrix theory, the global exponential robust stability of the system is studied. The obtained conditions containing diffusion terms improve the conservative of the previous results. An illustrated example shows how to use our results in practice.

Keywords: Hopfield neural networks, Reaction-diffusion, Delays, Robust stability.

1 Introduction

Hopfield neural networks (HNN), which were proposed by Hopfield [1] in 1982, caused a high degree of attention due to their potential applications for associative memory, parallel computation and their ability to solve difficult optimization problems [2]. Considering that time delays, which not only reduces the transmission speed of signal but also leads to instability of networks, unavoidably exist in neural networks, the stability of HNN with delays [3-7] has important theoretical significance and practical value. While neural networks model is only connected with time, it is a differential equation. However, strictly speaking, diffusion effect cannot be avoided in the neural networks when electrons are moving in asymmetric electromagnetic field, which could cause instability of the states, the stability analysis for delayed neural networks with reaction-diffusion terms becomes increasingly significant. Wang [8] considered dynamical behavior of HNN with time delays and reaction-diffusion terms, and got the L_2 -norm global exponential stability of HNN for the first time. Thereafter, large numbers of papers about neural networks with reaction-diffusion terms have emerged see [9-12] and reference therein. Nevertheless, the sufficient conditions for the stability of networks are the same as those obtained in the cases that there are not reaction-diffusion terms in the system because the reaction-diffusion terms are all eliminated by inequality analysis techniques in the above references, namely, these results are conservative. With Dirichlet boundary conditions, recurrent neural networks [13] and cellular neural networks [14] with reaction-diffusion terms are researched, the

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 693–701, 2010.

sufficient conditions including reaction-diffusion terms are less conservative, but the existence and uniqueness of the equilibrium point were not considered.

As far as we know, the entire existing criterion about the existence and uniqueness of the equilibrium point of neural networks do not contain reaction-diffusion terms. In this paper, with relative lemma of homeomorphism mapping and M-matrix theory, some conditions including reaction-diffusion terms are obtained to ensure the existence and uniqueness and global exponential stability of the equilibrium point of HNN with reaction-diffusion terms.

2 Model Description

Assuming that $\Omega \in R^m$ is a compact set with smooth boundary $\partial\Omega$, and $mes\Omega > 0$. Let $u_i(t,x)$ be the state of neural networks, $x \in \Omega$, $i = 1,2,...,n$. Denote by $\partial / \partial m$ the outward normal derivative. Denote $L^2(\Omega)$ the Lebesgue measurable function spaces, i.e. $\| u_i \|_{L^2} = (\int_\Omega | u_i |^2 dx)^{1/2}$, $i = 1,2,...,n$. For matrix $A = (a_{ij})_{n\times n}$, $|A|$ denotes the absolute-value matrix given by $|A| = (|a_{ij}|)_{n\times n}$; For vector $u = (u_1, u_2, ..., u_n) \in R^n$, $\| u \|$ denotes a vector norm defined by $\| u \| = (\sum_{i=1}^{n} \| u_i \|_{L^2}^2)^{\frac{1}{2}}$.

Hopfield neural networks with time delays and reaction-diffusion terms is described as follows

$$\frac{\partial u_i}{\partial t} = \sum_{k=1}^{m} \frac{\partial}{\partial x_k}(D_{ik} \frac{\partial u_i(t,x)}{\partial x_k}) - d_i(u_i(t,x)) + \sum_{j=1}^{n}[a_{ij} g_j(u_j(t,x)) + b_{ij} g_j(u_j(t-\tau_{ij}(t),x))] + J_i, \quad t \geq 0,\ i = 1,2,...,n,\ x \in \Omega, \tag{1a}$$

$$u_i(t,x)\big|_{\partial\Omega} = 0,\ i = 1,2,...,n,\ x \in \Omega. \tag{1b}$$

Where $n \geq 2$ denotes the number of neurons, $u_i(t,x)$ represents the state variable of ith neuron, $i = 1,2,...,n$. $D_{ik}(t,x) \geq 0$ denotes diffusion function and let $\theta_i = \min_{1\leq k\leq m}\{sup_{t\geq 0, x\in\Omega} D_{ik}\}$, $i = 1,2,...,n$. $d_i(u_i(t,x))$ is an appropriately behaved function, $A = (a_{ij})_{n\times n}$, $B = (b_{ij})_{n\times n}$, represent the weight matrices of the neurons, $g(u) = (g_1(u_1), g_2(u_2), ..., g_n(u_n))^T$ corresponds to the activation function of neurons. $\tau_{ij}(t) \geq 0$, $i, j = 1,2,...,n$, is time-varying delays of the neural networks, and $\tau = sup_{1\leq i,j\leq n, t\in R}\{\tau_{ij}(t)\}$. J_i denotes the external input on the ith neuron, let $J = (J_1, J_2, ..., J_n)^T$. The initial conditions of Eq. (1) are of the forms $u_i(s,x) = \phi_i(s,x)$, $s \leq 0$, where ϕ_i is bounded and continuous on $[-\tau, 0]$, $i = 1,2,...,n$.

Define interval matrix as follows:

$$A_I = \{A = (a_{ij})_{n\times n} : \underline{A} \leq A \leq \overline{A} \quad i.e.\ \underline{a}_{ij} \leq a_{ij} \leq \overline{a}_{ij}, i, j = 1,2,...,n\};$$

$$B_I = \{B = (b_{ij})_{n\times n} : \underline{B} \leq B \leq \overline{B} \quad i.e.\ \underline{b}_{ij} \leq b_{ij} \leq \overline{b}_{ij}, i, j = 1,2,...,n\},$$

let $A^0=(a_{ij}^0)_{n\times n}$, $B^0=(b_{ij}^0)_{n\times n}$, where $a_{ij}^0=\max\{|\underline{a}_{ij}|,|\overline{a}_{ij}|\}$, $b_{ij}^0=\max\{|\underline{b}_{ij}|,|\overline{b}_{ij}|\}$, $i,j=1,2,...,n$. Let $u^*=(u_1^*,u_2^*,\cdots,u_n^*)^T$ be the equilibrium point of system (1).

Definition 1. The equilibrium point of system (1) is said to be globally exponentially robustly stable if there exist constant $M>0$ and $\lambda>0$ such that for all $A\in A_I$, $B\in B_I$ and $J\in R^n$, $\|u(t,x)-u^*\|\le M\|\phi(s,x)-u^*\|e^{-\lambda t}$ holds, where $t\ge 0$, $x\in\Omega$, $\|\phi(s,x)-u^*\|=\sup\limits_{s\in[-\tau,0]}\sum\limits_{i=1}^{n}\|\phi_i(s,x)-u_i^*\|_{L_2}$.

Definition 2 [7]**.** A real matrix $A=(a_{ij})_{n\times n}$ is said to be a M-matrix if $a_{ij}\le 0$, $i,j=1,2,...,n$, $i\ne j$, and all successive principal minors of A are positive.

In the following, a lemma given in [14] is proposed, but its form is different from the lemma in [14].

Lemma 1 [3]**.** Let $\Omega\in R^m$ is a compact set, which has smooth boundary $\partial\Omega$. Let $x\in\Omega$, $|x_k|\le\omega_k$ ($k=1,2,...,m$), and $u_i(x)$ be a real-valued function belonging to $C^1(\Omega)$ and $u_i(x)\big|_{\partial\Omega}=0$, then $\int_\Omega u_i^2dx\le\frac{\pi^2}{m}\int_\Omega\nabla u_i^T\nabla u_i dx$, where $\pi=\max\limits_{1\le k\le m}\omega_k$, $i=1,2,...,n$.

Assumption A1. For each $i\in\{1,2,...,n\}$, function $d_i:R\to R$, is strictly monotone increasing, i.e. there exists constant $\gamma_i>0$ such that for all $u_i\ne v_i$,

$$0<\gamma_i\le\frac{d_i(u_i)-d_i(v_i)}{u_i-v_i}.$$

Assumption A2. Each function $g_i:R\to R$, $i=1,2,...,n$, is globally Lipschitz with Lipschitz constant L_i, i.e. for all u_i, v_i, $|g_i(u_i)-g_i(v_i)|\le L_i|u_i-v_i|$.

Define $H(u)$ is a nonlinear map associated with Eq.(1).

Lemma 2. If $H(u)\in C^0$ satifies (i) $H(u)$ is injective on R^n; (ii) $\|H(u)\|\to\infty$ as $\|u\|\to\infty$, then $H(u)$ is the homeomorphism of R^n.

It is well known that if $H(u)$ is homeomorphism on R^n, then neural networks (1) have a uniqueness equilibrium point u^*. In the following section, we will give a theorem ensuring $H(x)$ is a homeomorphism.

Theorem 1. Suppose that **Assumption A1** and **Assumption A2** are satisfied, then neural networks (1) has uniqueness equilibrium point u^* if for all $A\in A_I$, $B\in B_I$, $J\in R^n$, matrix P is a M-matrix, here

$$p_{ii}=\gamma_i+\theta_i\frac{m}{\pi^2}-\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}^0|+|b_{ij}^0|)L_j;\ p_{ij}=-\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}^0|+|b_{ij}^0|)L_j,\ i=1,2,...,n.$$

Proof. For all $A\in A_I$, $B\in B_I$, due to P is a M-matrix, from the property of M-matrix [7], we know that there exist $\xi_i>0$, $i=1,2,...,n$, such that

$$\xi_i[\frac{m}{\pi^2}\theta_i+\gamma_i-\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}^0|+|b_{ij}^0|)L_j]-\frac{1}{2}\sum_{j=1}^{n}\xi_j(|a_{ji}^0|+|b_{ji}^0|)L_j>0,$$

then for a sufficient small $\delta>0$, we have

$$\xi_i[\frac{m}{\pi^2}\theta_i+\gamma_i-\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}|+|b_{ij}|)L_j]-\frac{1}{2}\sum_{j=1}^{n}\xi_j(|a_{ji}|+|b_{ji}|)L_j\geq\delta>0. \quad (2)$$

Consider a map $H(u)=(H_1(u),H_2(u),...,H_n(u))^T\in C^0(R^n,R^n)$ endowed with the norm $\|\cdot\|=(\sum_{i=1}^{n}\|\cdot\|_{L^2}^2)^{\frac{1}{2}}$, where

$$H_i(u)=\sum_{k=1}^{m}\frac{\partial}{\partial x_k}[D_{ik}\frac{\partial u_i}{\partial x_k}]-d_i(u_i)+\sum_{j=1}^{n}(a_{ij}+b_{ij})g_j(u_j)+J_i,\ i=1,2,...,n. \quad (3)$$

First of all, we demonstrate the injective part.

When $H(u)=H(v)$, we get the following equation:

$$\sum_{k=1}^{m}\frac{\partial}{\partial x_k}(D_{ik}\frac{\partial u_i}{\partial x_k})-d_i(u_i)+\sum_{j=1}^{n}(a_{ij}+b_{ij})g_j(u_j)=\sum_{k=1}^{m}\frac{\partial}{\partial x_k}(D_{ik}\frac{\partial v_i}{\partial x_k})-d_i(v_i)+\sum_{j=1}^{n}(a_{ij}+b_{ij})g_j(v_j)$$

Considering **Assumption A1**, for $i=1,2,...,n$, we get

$$\sum_{k=1}^{m}\frac{\partial}{\partial x_k}[D_{ik}\frac{\partial(u_i-v_i)}{\partial x_k}]-\gamma_i|u_i-v_i|+\sum_{j=1}^{n}(a_{ij}+b_{ij})[g_j(u_j)-g_j(v_j)]\geq 0. \quad (4)$$

Multiply both sides of (4) by $|u_i-v_i|$ and integrate it on the domain Ω with respect to x, and consider **Assumption A2**, we get

$$\begin{aligned}&\int_\Omega\gamma_i|u_i-v_i|^2dx\\ &\leq\int_\Omega|u_i-v_i|\sum_{k=1}^{m}\frac{\partial}{\partial x_k}[D_{ik}\frac{\partial(u_i-v_i)}{\partial x_k}]dx+\int_\Omega\sum_{j=1}^{n}(a_{ij}+b_{ij})|u_i-v_i|(g_j(u_j)-g_j(v_j))dx\\ &\leq-\theta_i\frac{m}{\pi^2}\int_\Omega|u_i-v_i|^2dx+\int_\Omega\sum_{j=1}^{n}(|a_{ij}|+|b_{ij}|)L_j|u_i-v_i||u_j-v_j|dx\end{aligned} \quad (5)$$

Estimation (5) through Holder inequality and Young inequality leads to

$$\begin{aligned}&\int_\Omega\gamma_i|u_i-v_i|^2dx\\ &\leq-\theta_i\frac{m}{\pi^2}\int_\Omega|u_i-v_i|^2dx+\sum_{j=1}^{n}(|a_{ij}|+|b_{ij}|)L_j(\int_\Omega|u_i-v_i|^2dx)^{\frac{1}{2}}(\int_\Omega|u_j-v_j|^2dx)^{\frac{1}{2}}\\ &\leq-[\theta_i\frac{m}{\pi^2}-\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}|+|b_{ij}|)L_j]\int_\Omega|u_i-v_i|^2dx+\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}|+|b_{ij}|)L_j\int_\Omega|u_j-v_j|^2dx.\end{aligned}$$

Moreover

$$[\gamma_i+\theta_i\frac{m}{\pi^2}-\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}|+|b_{ij}|)L_j]\|u_i-v_i\|_{L^2}^2-\frac{1}{2}\sum_{j=1}^{n}(|a_{ij}|+|b_{ij}|)L_j\|u_j-v_j\|_{L^2}^2\leq 0,$$

namely $P(\|u_1-v_1\|_{L^2}^2,\|u_2-v_2\|_{L^2}^2,...,\|u_n-v_n\|_{L^2}^2)^T\leq 0$.

Since P is a M matrix, from the property of M-matrix, it follows that $\| u_i - v_i \|_{L^2}^2 = 0$, namely $u_i = v_i$, $i = 1,2,...,n$. So the map $H(u) \in C^0$ is injective on R^n.

Next, we will demonstrate $\| H(u) \| \to \infty$ as $\| u \| \to \infty$.

Let $\overline{H_i(u)} = H_i(u) - H_i(0)$, where $H_i(0) = \sum_{j=1}^{n} (a_{ij} + b_{ij}) g_j(0) + J_i$, *i.e.*

$$\overline{H_i(u)} = \sum_{l=1}^{m} \frac{\partial}{\partial x_l} (D_{il} \frac{\partial u_i}{\partial x_l}) - d_i(u_i) + \sum_{j=1}^{n} (a_{ij} + b_{ij})[g_j(u_j) - g_j(0)], \ i = 1,2,...,n. \quad (6)$$

It is suffice to show $\| \overline{H(u)} \| \to \infty$ as $\| u \| \to \infty$ in order to demonstrate the property $\| H(u) \| \to \infty$ as $\| u \| \to \infty$. Multiply both sides of Eq.(6) by u_i and integrate them on the domain Ω with respect to x, we get

$$\int_\Omega \overline{H_i(u)} u_i dx = \int_\Omega \{ u_i \sum_{k=1}^{m} \frac{\partial}{\partial x_k} (D_{ik} \frac{\partial u_i}{\partial x_k}) - \gamma_i u_i^2 + u_i \sum_{j=1}^{n} (a_{ij} + b_{ij})[g_j(u_j) - g_j(0)] \} dx. \quad (7)$$

From **Assumption 1** and in virtue of Holder inequality and Young inequality, we have

$$\int_\Omega \overline{H_i(u)} u_i dx \le -(\frac{m}{\pi^2} \theta_i + \gamma_i) \int_\Omega |u_i|^2 dx + \sum_{j=1}^{n} (|a_{ij}| + |b_{ij}|) L_j (\int_\Omega |u_i|^2 dx)^{\frac{1}{2}} (\int_\Omega |u_j|^2 dx)^{\frac{1}{2}}$$

$$\le -[\frac{m}{\pi^2} \theta_i + \gamma_i - \frac{1}{2} \sum_{j=1}^{n} (|a_{ij}| + |b_{ij}|) L_j] \int_\Omega |u_i|^2 dx + \frac{1}{2} \sum_{j=1}^{n} (|a_{ij}| + |b_{ij}|) L_j \int_\Omega |u_j|^2 dx. \quad (8)$$

Multiply both sides of (8) by ξ_i and consider (2), we get

$$\sum_{i=1}^{n} \xi_i \int_\Omega \overline{H_i(u)} u_i dx$$

$$\le \sum_{i=1}^{n} \xi_i \{ -[\frac{m}{\pi^2} \theta_i + \gamma_i - \frac{1}{2} \sum_{j=1}^{n} (|a_{ij}| + |b_{ij}|) L_j] \int_\Omega |u_i|^2 dx + \frac{1}{2} \sum_{j=1}^{n} (|a_{ij}| + |b_{ij}|) L_j \int_\Omega |u_j|^2 dx \}$$

$$\le -\delta \sum_{i=1}^{n} \| u_i \|_{L^2}^2.$$

Furthermore $\delta \sum_{i=1}^{n} \| u_i \|_{L^2}^2 \le -\sum_{i=1}^{n} \xi_i \int_\Omega \overline{H_i}(u) u_i dx \le \max_{1 \le i \le n} \{\xi_i\} \sum_{i=1}^{n} \| \overline{H_i(u)} \|_{L^2} \| u_i \|_{L^2}$. (9)

Employing Holder inequality for (9), we obtain

$$\delta \| u \|^2 \le \max_{1 \le i \le n} \{\xi_i\} (\sum_{i=1}^{n} \| \overline{H_i(u)} \|_{L^2}^2)^{\frac{1}{2}} (\sum_{i=1}^{n} \| u_i \|_{L^2}^2)^{\frac{1}{2}} = \max_{1 \le i \le n} \{\xi_i\} \| \overline{H(u)} \| . \| u \|,$$

i.e. $\delta \| u \| \le \max_{1 \le i \le n} \{\xi_i\} \| \overline{H(u)} \|$. Apparently, $\| \overline{H(u)} \| \to \infty$ as $\| u \| \to \infty$, which directly implies that $\| H(u) \| \to \infty$ as $\| u \| \to \infty$. It can be concluded that $H(u)$ is a homeomorphism on R^n, namely, Eq. (1) has unique equilibrium point. □

3 Main Results

In this section, we will establish a family of conditions ensuring global exponential robust stability of Eq. (1).

Conveniently, we introduce coordinate translation for model (1), let $z_i = u_i - u_i^*$, $i = 1,2,...,n$, the system (1) can be rewritten as follows:

$$\frac{\partial z_i}{\partial t} = \sum_{k=1}^{m} \frac{\partial}{\partial x_k}[D_{ik}(t,x)\frac{\partial z_i}{\partial x_k}] - h_i(z_i(t,x)) + \sum_{j=1}^{n}[a_{ij}f_j(z_j(t,x)) + b_{ij}f_j(z_j(t-\tau_{ij}(t),x))]\},\ t \geq 0,\ i = 1,2,...,n,\ x \in \Omega, \tag{10a}$$

$$z_i(t,x)\big|_{\partial\Omega} = 0,\ i = 1,2,...,n, \tag{10b}$$

where $h_i(z_i(t,x)) = d_i(z_i(t,x) + u_i^*) - d_i(u_i^*)$, $f_j(z_j(t,x)) = g_j(z_j(t,x) + u_j^*) - g_j(u_j^*)$.

The initial condition of Eq. (10) is $\varphi_i(s,x) = \phi_i(s,x) - u_i^*$, $-\tau \leq s \leq 0$, $i = 1,2,...,n$. Next, we will give some sufficient conditions for Eq.(10).

Theorem 2. Suppose that **Assumption A1** and **Assumption A2** are satisfied, then the zero solution of system (10) is globally exponentially robustly stable if for all $A \in A_I$, $B \in B_I$, matrix P is a M-matrix, where

$$p_{ii} = \gamma_i + \theta_i \frac{m}{\pi^2} - \frac{1}{2}\sum_{j=1}^{n}(a_{ij}^0 + b_{ij}^0)L_j;\ p_{ij} = -\frac{1}{2}\sum_{j=1}^{n}(a_{ij}^0 + b_{ij}^0)L_j,\ i = 1,2,...,n.$$

Proof. Firstly, from (2), it can be concluded that

$$-\xi_i[\gamma_i + \theta_i \frac{m}{\pi^2} - \frac{1}{2}\sum_{j=1}^{n}(|a_{ij}| + |b_{ij}|)L_j] + \frac{1}{2}\sum_{j=1}^{n}\xi_j(|a_{ij}| + |b_{ij}|)L_j < 0,\ i = 1,2,...,n.$$

Constructing the functions

$$F_i(\mu) = -\xi_i[-\mu + \gamma_i + \theta_i \frac{m}{\pi^2} - \frac{1}{2}\sum_{j=1}^{n}L_j(|a_{ij}| + |b_{ij}|)] + \frac{1}{2}\sum_{j=1}^{n}\xi_j(|a_{ij}| + e^{\mu\tau}|b_{ij}|)L_j.$$

Obviously $F_i(0) < 0$. Since $F_i(\mu)$, $i = 1,2,...,n$, are continuous functions, so there exists a constant $\lambda > 0$, such that $F_i(\lambda) < 0$, *i.e.*

$$F_i(\lambda) = -\xi_i[-\lambda + \gamma_i + \theta_i \frac{m}{\pi^2} - \frac{1}{2}\sum_{j=1}^{n}L_j(|a_{ij}| + |b_{ij}|)] + \frac{1}{2}\sum_{j=1}^{n}\xi_j(|a_{ij}| + e^{\lambda\tau}|b_{ij}|)L_j < 0 \tag{11}$$

Let $V_i(t) = e^{\lambda t}\| z_i(t,x)\|_{L^2}^2$, $i = 1,2,...,n$, calculating the upper right derivation D^+V_i of V_i along the solution of Eq.(10), we get

$$D^+V_i(t) = \lambda e^{\lambda t}\| z_i(t,x)\|_{L^2}^2 + e^{\lambda t}\int_\Omega z_i(t,x)\{\sum_{k=1}^{m}\frac{\partial}{\partial x_k}[D_{ik}(x)\frac{\partial z_i(t,x)}{\partial x_k}] - h_i(z_i(t,x)) + \sum_{j=1}^{n}[a_{ij}f_j(z_j(t,x)) + b_{ij}f_j(z_j(t-\tau_{ij}(t),x))]\}dx,\ i = 1,2,...,n, \tag{12}$$

By the **Lemma 1**, we have

$$\int_\Omega z_i(t,x)\sum_{k=1}^{m}\frac{\partial}{\partial x_k}[D_{ik}\frac{\partial z_i(t,x)}{\partial x_k}]dx \le -\theta_i\int_\Omega \nabla z_i^T(t,x)\nabla z_i(t,x)dx \le -\theta_i\frac{m}{\pi^2}\int_\Omega z_i^2(t,x)dx \quad (13)$$

Substituting (13) into (12), and from **AssumptionsA1-A2**, we have

$$D^+V_i(t) \le \lambda e^{\lambda t}\| z_i(t,x)\|_{L^2}^2 -\theta_i\frac{m}{\pi^2}e^{\lambda t}\int_\Omega z_i^2(t,x)dx + e^{\lambda t}\int_\Omega\{-\gamma_i z_i^2(t,x)$$
$$+\sum_{j=1}^{n}L_j[|z_i(t,x)|.|a_{ij}|.|z_j(t,x)|+|z_i(t,x)|.|b_{ij}|.|z_j(t-\tau_{ij}(t),x)|]\}dx, \quad (14)$$

by using Holder inequality and Young inequality for (14), we get

$$D^+V_i(t) \le \lambda e^{\lambda t}\| z_i(t,x)\|_{L^2}^2 -\theta_i\frac{m}{\pi^2}e^{\lambda t}\| z_i(t,x)\|_{L^2}^2 -\gamma_i e^{\lambda t}\| z_i(t,x)\|_{L^2}^2$$
$$+\frac{1}{2}e^{\lambda t}\sum_{j=1}^{n}L_j|a_{ij}|(\| z_i(t,x)\|_{L^2}^2+\| z_j(t,x)\|_{L^2}^2)+\frac{1}{2}e^{\lambda t}\sum_{j=1}^{n}L_j|b_{ij}|[\| z_i(t,x)\|_{L^2}^2+\| z_j(t-\tau_{ij}(t),x)\|_{L^2}^2)]$$
$$\le[\lambda-\theta_i\frac{m}{\pi^2}-\gamma_i+\frac{1}{2}\sum_{j=1}^{n}L_j(|a_{ij}|+|b_{ij}|)]V_i(t)+\frac{1}{2}\sum_{j=1}^{n}L_j[|a_{ij}|V_j(t)+|b_{ij}|e^{\lambda\tau}\sup_{t-\tau\le s\le t}V_j(s)],$$
$$i=1,2,...,n. \quad (15)$$

Defining the curve, $\zeta=\{y(\chi):y_i=\xi_i\chi,\chi>0\}$, and the sets $\kappa(y)=\{z:0\le z\le y, y\in\zeta\}$. Let $\xi_{\min}=min_{1\le i\le n}\{\xi_i\}$, $\xi_{\max}=max_{1\le i\le n}\{\xi_i\}$. Taking $\chi_0=\delta\|\varphi\|^2/\xi_{min}$, here $\delta>1$ is a constant. Then $\{V:V=\mathrm{e}^{\lambda s}\|\varphi\|^2,-\tau\le s\le 0\}\subset\kappa(y_0(\chi_0))$, *i.e.* $V_i(s)<\xi_i\chi_0$, $-\tau\le s\le 0$, $i=1,2,...,n$.

We claim that $V_i(t)<\xi_i\chi_0$, $t\ge 0$, $i=1,2,...,n$. If it is not true, then there exists some i and $t'>0$, such that $V_i(t')=\xi_i\chi_0$, $D^+V_i(t')\ge 0$ and $V_j(t)\le\xi_j\chi_0$, $-\tau\le t\le t'$, $j=1,2,...,n$. However, from (11) and (15), we get

$$D^+V_i(t') \le [\lambda-\theta_i\frac{m}{\pi^2}-\gamma_i+\frac{1}{2}\sum_{j=1}^{n}L_j(|a_{ij}|+|b_{ij}|)]V_i(t')+\frac{1}{2}\sum_{j=1}^{n}[L_j(|a_{ij}|V_j(t')+|b_{ij}|V_j(t'-s))]$$
$$\le[\lambda-\theta_i\frac{m}{\pi^2}-\gamma_i+\frac{1}{2}\sum_{j=1}^{n}L_j(|a_{ij}|+|b_{ij}|)]\xi_i\chi_0+\frac{1}{2}\sum_{j=1}^{n}L_j(|a_{ij}|+e^{\lambda\tau}|b_{ij}|)\xi_j\chi_0<0$$

This is a contradiction. So $V_i(t)<\xi_i\chi_0$, $t\ge 0$, $i=1,2,...,n$, *i.e.* $\| z_i(t,x)\|_{L^2}^2<e^{-\lambda t}\xi_i\chi_0$, namely, $\| z(t,x)\|_{L^2}<e^{\frac{-\lambda t}{2}}(\xi_i\chi_0)^{1/2}\le M\|\varphi\|e^{\frac{-\lambda t}{2}}$, here $M=(\xi_{max}\delta/\xi_{min})^{1/2}$.

From **Definition 1**, the zero solution of the system (10) is globally exponentially robustly stable, namely, the equilibrium point of Eq. (1) is globally exponentially robustly stable. The proof is completed. □

4 Example

Consider the following system:

$$\frac{\partial u_i}{\partial t} = \frac{\partial}{\partial x}[D_i(x)\frac{\partial u_i}{\partial x}] - d_i(u_i(t,x)) + \{\sum_{j=1}^{2}[a_{ij}g_j(u_j(t,x)) + b_{ij}g_j(u_j(t-\tau_{ij}(t),x))] + J_i\}, \quad (16)$$

with the boundry condition $u_i(t,x)\big|_{\partial\Omega} = 0$, taking $\tau_{ij}(t) = 1 + 0.6\sin t$, $J_i = 0$ where $i, j = 1,2$. Let matrix $A = (a_{ij})_{2\times 2} = \begin{bmatrix} 1 & -0.2 \\ 0.5 & -1 \end{bmatrix}$, $B = (b_{ij})_{2\times 2} = \begin{bmatrix} 0 & 1 \\ -0.4 & 0 \end{bmatrix}$, and $D(x) = \begin{bmatrix} 2+\sin x_2 & 5 & 4+\sin^2 x_1 \\ 6.5 & 3+\cos^2 x_3 & 2+|x_1| \end{bmatrix} > 0$, $|x_k| < 2$ $(1 \le k \le 3)$, $d_1(u_1(t,x)) = 5u_1(t,x)$, $d_2(u_2(t,x)) = 4.5u_2(t,x)$, $g_j(u_j) = \frac{e^{u_j} - e^{-u_j}}{e^{u_j} + e^{-u_j}} = \tanh u_j$. Obviously, the **Assumptions A1-A2** hold and by easy computation, we get $\theta_1 = 1$, $\theta_2 = 2$, $\pi = 2$, $\gamma_1 = 5$, $\gamma_2 = 4.5$; $L = diag(1,1)$; $\tau = 1.6$. Furthermore, we can obtain matrix $P = \begin{bmatrix} 4.65 & -1.1 \\ -0.55 & 4.95 \end{bmatrix}$.

Clearly, from **Definition 2**, we know that *P* is M-matrix, so the equilibrium point of system (16) is globally exponentially stable.

5 Conclusion

In this paper, applying vector Lyapunov method and M-matrix theory, the analysis of existence, uniqueness and global exponential robust stability of the equilibrium point of a class of delayed Hopfield neural networks with reaction-diffusion terms have been presented. The obtained sufficient conditions are less conservative. An example is given at last to illustrate the practicability of our results.

Acknowledgments. This work is supported by Natural Science Foundation of China (No. 10772152, 60974132), Doctoral Fund of Ministry of Education of China (No. 200806130003).

References

1. Hopfield, J.J.: Neural Networks and Physical Systems with Emergent Collective Abilities. In: Proceedings of the National Academy of Sciences, pp. 2554–2558 (1982)
2. Tank, D.W., Hopfield, J.J.: Simple 'neural' Optimization Networks: An A /D Converter, Signal Decision Circuit, and a Linear Programming Network. J. IEEE Trans. Circuits and Systems 33, 533–541 (1986)
3. Zhang, J.Y.: Absolutely Stability of a Class of Neural Networks with Unbounded Delay. J. International Journal of Circuit Theory and Applications 32, 11–21 (2004)
4. Tan, X.H., Zhang, J.Y., Yang, Y.R.: Global Exponential Stability of Hopfield Neural Networks. J. Southwest Jiaotong University 40, 338–342 (2005)

5. Arik, S., Orman, Z.: Global Stability Analysis of Cohen-Grossberg Neural Networks with Time Varying Delays. J. Phys. Lett. A. 341, 410–421 (2005)
6. Arki, S., Tavsanoglu, V.: Global Asymptotic Stability Analysis of Bidirectional Associative Memory Neural Networks with Constant Time Delays. J. Neurocomputing 68, 161–176 (2005)
7. Zhang, J.Y., Suda, Y., Iwasa, T.: Absolutely Exponential Stability of a Class of Neural Networks with Unbounded Delay. J. Neural Networks 17, 391–397 (2004)
8. Wang, L.S., Xu, D.Y.: Global Exponential Stability of Hopfield Neural Networks with Variable Time Delays and Reaction-Diffusion Terms. J. Science in China (Series E) 33, 488–495 (2003)
9. Allegretto, W., Papini, D.: Stability for Delayed Reaction-Diffusion Neural Networks. J. Physics Letters A 360, 669–680 (2007)
10. Li, K.L., Song, Q.K.: Exponential Stability of Impulsive Cohen-Grossberg Neural Networks with Time-Varying Delays and Reaction-Diffusion Terms. Neurocomputing 72, 231–240 (2008)
11. Wang, L., Gao, Y.: Global Exponential Robust Stability of Reaction-Diffusion Interval Neural Networks with Time-Varying Delays. J. Physics Letters A 350, 342–348 (2006)
12. Liang, J., Cao, J.: Global Exponential Stability of Reaction-Diffusion Recurrent Neural Networks with Time-Varying Delays. J. Physics Letters A 314, 434–442 (2003)
13. Zhao, H.Y., Wang, K.L.: Dynamical Behaviors of Cohen-Grossberg Neural Networks with Delays and Reaction-Diffusion Terms. J. Neurocomputing 70, 536–543 (2006)
14. Lu, J.: Global Exponential Stability and Periodicity of Reaction-Diffusion Delayed Recurrent Neural Networks with Dirichlet Boundary Conditions. J. Chaos, Solitons and Fractals 35, 116–125 (2008)
15. Wang, J., Lu, J.: Global Exponential Stability of Fuzzy Cellular Neural Networks with Delays and Reaction-Diffusion Terms. J. Chaos, Solitons and Fractals 38, 878–885 (2008)

Stability and Bifurcation of a Three-Dimension Discrete Neural Network Model with Delay

Wei Yang and Chunrui Zhang*

Department of Mathematics, Northeast Forestry University, Harbin 150040, P.R. China
angelyw2008, math@nefu.edu.cn

Abstract. In this paper, we consider a three-dimension discrete neural network model with delay. The characteristic equation of the linearized system at the zero solution is a polynomial equation involving very high order terms. We derive some sufficient and necessary conditions on the asymptotic stability of the zero solution. In addition, it is found that there exist Hopf bifurcations when the parameter passes a sequence of critical values by using Extensional Jury Criterion. Finally, computer simulations are performed to support the theoretical predictions.

Keywords: Discrete neural network, Stability, Hopf bifurcation, Periodic solution, Jury Criterion.

1 Introduction

Since Hopfield's work, a great deal of research activities into the theories and applications of neural networks introduced by Hopfield. Recently, the dynamical behaviors (including stability, instability, periodic oscillatory and chaos) of the discrete-time Hopfield neural networks without or with delay have received increasing interesting due to their promising potential applications in many fields. Some results have been reported. See [1]~[8]. For example, Guo et al.[1] discuss the linear stability and Hopf bifurcation of a discrete system of two neurons with three delays. C.Zhang and B.Zheng [2] consider a two-dimension discrete neural network model with multi-delay by using Euler method. However, no one has applied Extensional Jury Criterion[9] to discuss the dynamical behaviors of a discrete neural network model without or with delay.

In this paper, we consider a three-dimension discrete neural network model with delay:

$$\begin{cases} x_{n+1} = ax_n + \alpha f(x_{n-k}) + \beta g(y_{n-k}) + \beta g(z_{n-k}) \\ y_{n+1} = ay_n + \alpha f(y_{n-k}) + \beta g(z_{n-k}) + \beta g(x_{n-k}) \\ z_{n+1} = az_n + \alpha f(z_{n-k}) + \beta g(x_{n-k}) + \beta g(y_{n-k}) \end{cases} \tag{1.1}$$

* Corresponding author.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 702–710, 2010.

where x_n, y_n, z_n denote the activations of three neurons, α、β、k、a are parameters. $a > 0$ is the internal decay of the neurons, k is non-negative integer which denote the synaptic transmission delay, α、β are the connection weights. f、$g : R \to R$ is the activation function. We allow that every neuron has self-feedback and delayed signal transmission which is due to the finite switching speed of neurons.

The goal of this paper is to investigate how parameter affects the discrete neural network model with delay (1.1) by using Hopf bifurcation theory of discrete system and Extensional Jury Criterion [9]. We not only discuss the stability of the fixed point and the existence of the Hopf bifurcations, but also the direction of Hopf bifurcation and the stability of the bifurcation periodic solutions.

2 Stability Analysis and Bifurcation

Throughout this section, to establish the main results for the discrete neural network model with delay (1.1), we make the following hypothesis on the activation functions in (1.1) :

(H_1): f、$g : R \to R$ is a C^1 – smooth function with $f(0) = 0, g(0) = 0$;

(H_2): $f'(0) = g'(0) = 1$.

Under the assumptions (H_1) and (H_2), it follows that the origin (0,0,0) is an equilibrium of (1.1). Linearizing system(1.1) about origin (0,0,0) gives the following linear system:

$$\begin{cases} x_{n+1} = ax_n + \alpha x_{n-k} + \beta y_{n-k} + \beta z_{n-k} \\ y_{n+1} = ay_n + \alpha y_{n-k} + \beta z_{n-k} + \beta x_{n-k} \\ z_{n+1} = az_n + \alpha z_{n-k} + \beta x_{n-k} + \beta y_{n-k} \end{cases} \tag{2.1}$$

we can induce the method from [2] to mark $M_{n+1} = AM_n$

where

$$M_n = \left(x_n, x_{n-1}, x_{n-2}, \cdots, x_{n-k+1}, y_n, y_{n-1}, y_{n-2}, \cdots, y_{n-k+1}, z_n, z_{n-1}, z_{n-2}, \cdots, z_{n-k+1}\right)^T$$

$$A = \begin{bmatrix} R_1 & R_2 & R_2 \\ R_2 & R_1 & R_2 \\ R_2 & R_2 & R_1 \end{bmatrix}_{(3k+3)\times(3k+3)}$$

where

$$R_1 = \begin{bmatrix} a & 0 & \cdots & \cdots & 0 & \alpha \\ 1 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix}_{(k+1)\times(k+1)} \quad R_2 = \begin{bmatrix} 0 & 0 & \cdots & \cdots & 0 & \beta \\ 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}_{(k+1)\times(k+1)}$$

then, the characteristic equation for (2.1) is given by

$$\left|\lambda E - A\right| = (\lambda^{k+1} - a\lambda^k - \alpha)^3 - 3\beta^2(\lambda^{k+1} - a\lambda^k - \alpha) - 2\beta^3 = 0 \tag{2.2}$$

That is

$$(\lambda^{k+1} - a\lambda^k - \alpha - 2\beta)(\lambda^{k+1} - a\lambda^k - \alpha + \beta)^2 = 0 \tag{2.3}$$

For discussing the linear stability and Hopf bifurcations of the (1.1), we only need to analyze the distribution of the roots of the (2.4) and (2.5).

We consider the following two cases:

Case 1:

$$\lambda^{k+1} - a\lambda^k - \alpha - 2\beta = 0 \tag{2.4}$$

To work up simply, we can mark $a_{01} = -\alpha - 2\beta$.

Case 2:

$$(\lambda^{k+1} - a\lambda^k - \alpha + \beta)^2 = 0 \tag{2.5}$$

Same as a_{01} , we also mark $a_{02} = -\alpha + \beta$, also we can mark (2.4) and (2.5) as

$$\lambda^{k+1} - a\lambda^k + a_{0t} = 0, \quad t = 1,2 \tag{2.6}$$

It is well known that the trivial solution of (1.1) is asymptotically stable if all roots λ of (2.3) are inside the unit circle. Hence, to discuss the stability for (1.1), we give the following result by concerning Jury Criterion:

Theorem 1. If

$$(a^{'})\ \ a_{0t} > a - 1;\ (b^{'})\ (-1)^{k+1} a_{0t} > -1 - a;\ (c^{'})\ -1 < a_{0t} < 1;$$
$$(d^{'})\ \ \left|a_{0t}^{(2)}\right| > \left|a_{kt}^{(2)}\right|,\ \left|a_{0t}^{(3)}\right| > \left|a_{(k-1)t}^{(3)}\right|,\ \cdots\cdots,\ \left|a_{0t}^{(k-1)}\right| > \left|a_{3t}^{(k-1)}\right|,\ \left|a_{0t}^{(k)}\right| > \left|a_{2t}^{(k)}\right|.$$

are satisfied for (1.1), then the zero solution of (1.1) is asymptotically stable.

Where $t = 1,2; a_{jt}^{(1)} = a_{jt}, j = 0,1,\cdots,k+1,$

$$a_{jt}^{(i)} = \begin{vmatrix} a_{0t}^{(i-1)} & a_{[k-(i+j)+3]t}^{(i-1)} \\ a_{(k-i+3)t}^{(i-1)} & a_{jt}^{(i-1)} \end{vmatrix}, i = 2,3,\cdots,k+2,\ j = 0,1,2,\cdots,k-i+3.$$

Proof. $(a^{'})$. First, we can mark (2.3) as

$$D(\lambda; a_{0t}, a_{n-1}, a_n) = D(\lambda; a_{0t}, a_k, a_{k+1}) = a_{k+1}\lambda^{k+1} - a_k\lambda^k + a_{0t} \tag{2.7}$$

where $a_{k+1} = 1, a_k = a, a_{01} = -\alpha - 2\beta$, $a_{02} = -\alpha + \beta$ .By applying Jury Criterion , we have derived that $D(\lambda; a_{0t}, a_k, a_{k+1}) = 1 - a + a_{0t} > 0$. Hence, we have $a_{0t} > a - 1$;

($b^{'}$). We can substitute $\lambda = -1$ into (2.7),

$$(-1)^n D(\lambda; a_{0t}, a_k, a_{k+1}) = (-1)^{k+1}[(-1)^{k+1} - a(-1)^k + a_{0t}] > 0$$

then, we have

$$(-1)^{k+1} a_{0t} > -1 - a\ ;$$

($c^{'}$). We know that $|a_{0t}| < |a_{k+1}|$, so we have $-1 < a_{0t} < 1$;

($d^{'}$). By the table of [9] , it is easily to know

$$b_{0t} = a_{0t}^{(2)} \qquad c_{0t} = a_{0t}^{(3)} \qquad \cdots\cdots \qquad l_{0t} = a_{0t}^{(n-2)}$$

$$b_{1t} = a_{1t}^{(2)} \qquad c_{1t} = a_{1t}^{(3)} \qquad \cdots\cdots \qquad l_{1t} = a_{1t}^{(n-2)}$$

$$\cdots\cdots \qquad \cdots\cdots \qquad \cdots\cdots \qquad l_{2t} = a_{2t}^{(n-2)}$$

$$b_{(n-1)t} = a_{(n-1)t}^{(2)} \qquad c_{(n-2)t} = a_{(n-2)t}^{(3)} \qquad \cdots\cdots \qquad l_{3t} = a_{3t}^{(n-2)}$$

Since $n = k+1$ (2.7), we have

$$b_{(n-1)t} = a_{kt}^{(2)} \qquad c_{(n-2)t} = a_{(k-1)t}^{(2)} \qquad \cdots\cdots \qquad l_{3t} = a_{3t}^{(k-1)}$$

From Jury Criterion we know that

$$\left|a_{0t}^{(2)}\right| > \left|a_{kt}^{(2)}\right|, \left|a_{0t}^{(3)}\right| > \left|a_{(k-1)t}^{(3)}\right|, \cdots\cdots, \left|a_{0t}^{(k-1)}\right| > \left|a_{3t}^{(k-1)}\right|, \left|a_{0t}^{(k)}\right| > \left|a_{2t}^{(k)}\right|$$

The proof is completed.

Then, we discuss the bifurcation for the (1.1). We give the following result by concerning Extensional Jury Criterion[9] and the distribution of the zeros of polynomial [Zhang C. et al.,2006]. First, we give the Lemma 1:

Lemma 1[9] .(*Extensional Jury Criterion*) Define a real coefficient polynomial

$$D(z; a_{0,} a_1, \cdots, a_n), (n \geq 3, a_n > 0), \qquad (2.8)$$

The necessary and sufficient conditions for (2.8) to have a couple of conjugate complex roots and the length of modulo equal to 1, other length of modulo of $n-2$ roots are less than 1: when $n = 3$,the following conditions (i)(ii)(iii)(v) are established, when $n > 3$, (i)(ii)(iii)(iv)(v) are established:

(i) $D(1; a_0, a_1, \cdots, a_n) > 0$; (ii) $(-1)^n D(-1; a_0, a_1, \cdots, a_n) > 0$;

(iii) $|a_0| < |a_n|$; (iv) $|b_0| > |b_{n-1}|, |c_0| > |c_{n-2}|, \cdots, |l_0| > |l_{n-3}|$

(v) $m_0 = m_2$

By Lemmas 1 and the Hopf bifurcation theorem as stated in [10], we have the following results on stability and bifurcation in (1.1).

Theorem 2. If

$$(a^{'})\ \ a_{01} > a-1;\ \ (b^{'})\ (-1)^{k+1}a_{01} > -1-a;\ (c^{'})\ -1 < a_{01} < 1;$$
$$(d^{'})\ \ \left|a_{01}^{(2)}\right| > \left|a_{k1}^{(2)}\right|,\ \left|a_{01}^{(3)}\right| > \left|a_{(k-1)1}^{(3)}\right|,\ \cdots\cdots,\ \left|a_{01}^{(k-1)}\right| > \left|a_{31}^{(k-1)}\right|;$$
$$(e^{'})\ \ \left|a_{01}^{(k)}\right| = \left|a_{21}^{(k)}\right|.$$

are satisfied for (1.1), then (1.1) has a Generic Hopf bifurcation at the origin(0,0,0). Where $a_{j1}^{(1)} = a_{j1,}\, j = 0,1,\cdots,k+1,$

$$a_{j1}^{(i)} = \begin{vmatrix} a_{01}^{(i-1)} & a_{(k-(i+j)+3)1}^{(i-1)} \\ a_{(k-i+3)1}^{(i-1)} & a_{j1}^{(i-1)} \end{vmatrix}, i = 2,3,\cdots,k+2,\ j = 0,1,2,\cdots,k-i+3.$$

Proof. we can only to prove $(e^{'})$. From Lemma 1(v), we easily to know that $m_0 = m_2$ also mean $\left|a_{01}^{(k)}\right| = \left|a_{21}^{(k)}\right|$. The proof is completed.

The minimum absolute value of a_{01} in the inequality is marked a_{00}, and $e^{\pm iw_{00}}$ satisfies the following expression:

$$\begin{cases} \cos(k+1)\omega_{00} - a\cos k\omega_{00} + a_{00} = 0 \\ \sin(k+1)\omega_{00} - a\sin k\omega_{00} = 0 \end{cases} \tag{2.9}$$

Where $\omega \in [\frac{j\pi}{k}, \frac{\pi(j+1)}{k+1}]$, $j = 0,1,.....k-1$.

Since

$$\frac{d\left|\lambda(a_{00})\right|^2}{d(a_{00})} = \frac{-(k+1)\bar{\lambda}^{k+1}(a_{00}) + ak\bar{\lambda}^{k} - (k+1)\lambda^{k+1}(a_{00}) + ak\lambda^{k}}{[(k+1)\lambda^{k}(a_{00}) - ak\lambda^{k-1}(a_{00})][(k+1)\bar{\lambda}^{k}(a_{00}) - ak\bar{\lambda}^{k-1}(a_{00})]} \tag{2.10}$$

$$a = \frac{\sin(k+1)\omega_{00}}{\sin k\omega_{00}} \tag{2.11}$$

Substitute (2.11) into (2.10), we have

$$\left(\frac{d\left|\lambda(a_{00})\right|^2}{d\left|a_{00}\right|}\right)$$
$$= \frac{2(k+1)\sin\omega_{00} - 2\sin(k+1)\omega_{00}\cos(k+1)\omega_{00}}{\sin k\omega_{00}[(k+1)e^{ik\omega_{00}}(a_{00}) - ake^{i(k-1)\omega_{00}}(a_{00})][(k+1)e^{-ik\omega_{00}}(a_{00}) - ake^{-i(k-1)\omega_{00}}(a_{00})]}$$

where

$$\omega \in [\frac{j\pi}{k}, \frac{\pi(j+1)}{k+1}],\ j = 0,1,.....k-1.$$

Hence, we have obtained that $\sin k\omega_{00} > 0$. The sign of $\frac{d|\lambda(a_{00})|^2}{d|a_{00}|}$ is determined by $2(k+1)\sin\omega_{00} - 2\sin(k+1)\omega_{00}\cos(k+1)\omega_{00}$. In fact, we know $\cos(k+1)\omega_{00} > 0$ and $\sin(k+1)\omega_{00} < 0$, so $2(k+1)\sin\omega_{00} - \sin(k+1)\omega_{00}\cos(k+1)\omega_{00} > 0$. Finally, we have $\frac{d|\lambda(a_{00})|^2}{d|a_{00}|} > 0$. The proof is completed.

Theorem 3. If

(a) $a_{02} > a-1$; (b) $(-1)^{k+1}a_{02} > -1-a$; (c) $-1 < a_{02} < 1$;
(d) $\left|a_{02}^{(2)}\right| > \left|a_{k2}^{(2)}\right|$, $\left|a_{02}^{(3)}\right| > \left|a_{(k-1)2}^{(3)}\right|$, $\cdots\cdots$, $\left|a_{02}^{(k-1)}\right| > \left|a_{32}^{(k-1)}\right|$;
(e) $\left|a_{02}^{(k)}\right| = \left|a_{22}^{(k)}\right|$.

are satisfied for (1.1), then (1.1) has a Equivariant Hopf bifurcation at the origin(0,0,0). Where $a_{j2}^{(1)} = a_{j2}, j = 0,1,\cdots,k+1,$

$$a_{j2}^{(i)} = \begin{vmatrix} a_{02}^{(i-1)} & a_{(k-(i+j)+3)2}^{(i-1)} \\ a_{(k-i+3)2}^{(i-1)} & a_{j2}^{(i-1)} \end{vmatrix}, i = 2,3,\cdots,k+2,\ j = 0,1,2,\cdots,k-i+3.$$

Proof. The method of proof for Theorem 3 is the same as Theorem 2.

3 D_3-Equivariant and Multiple Bifurcations

Lemma 2. System (1.1) is D_3-equivalent, where D_3 is the dihedral group of order 6.
Proof: D_3 can be generated by matrices

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, Q = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}.$$

Hence, matrix group $D_3 = \{I_3, P, P^2, Q, QP, QP^2\}$. Consider the subgroup of D_3:

$$\Gamma_1 = \{P, \theta(\omega t) = \omega t - \frac{2\pi}{3}\}, \quad \Gamma_2 = \{Q, \theta(\omega t) = \omega t\}, \quad \Gamma_3 = \{QP, \theta(\omega t) = \omega t - \pi\}.$$

Let $T = \frac{2\pi}{\omega_{00}}$ and denote P_T the Banach space of all continuous T-periodic function $x(t)$. Denoting SP_T the subspace of P_T consisting of all T-periodic solution of system

(1.1). Let $\mathrm{Fix}(\Gamma, SP_T) = \{x \in SP_T; Gx = x, \text{for all } G \in D_3\}$,and using the Golubitshy's theorems [11], we have

$$\mathrm{DimFix}(\Gamma_1, SP_T) = 2, \quad \mathrm{DimFix}(\Gamma_2, SP_T) = 2, \quad \mathrm{DimFix}(\Gamma_3, SP_T) = 2$$

Theorem 4. Assume that a_{02} satisfy the conditions(a)(b)(c)(d)(e). Then near a_{02} there exist eight branches of small-amplitude periodic solutions of (1.1) with period T near $\dfrac{2\pi}{\omega_{00}}$ and they are:

①two phase-locked oscillations : $x_i(n) = x_{i-1}(n \pm \dfrac{2\pi}{3\omega_{00}})$ for i (mod) 3,

②three mirror-reflecting waves: $x_i(n) = x_j(n) \neq x_k(n)$ for some distinct *i, j, k* in {1, 2, 3},

③three standing waves: $x_i(n) = x_j(n \pm \dfrac{\pi}{\omega_{00}})$ for some distinct *i, j* in {1, 2, 3}.

4 Numerical Simulation

In this section, we will study the specific example of (1.1) along with numerical simulation. First we can use the numerical simulation to show the zero solution of (1.1) is asymptotically stable. Let $k = 2$, $f(x) = g(x) = \tanh$, $a = 1$, $\alpha = 0.01$, $\beta = -0.5$.

The characteristic equation is deprived as

$$(\lambda^3 - a\lambda^2 - \alpha - 2\beta)(\lambda^3 - a\lambda^2 - \alpha + \beta)^2 = 0$$

The result of numerical simulation as follow:

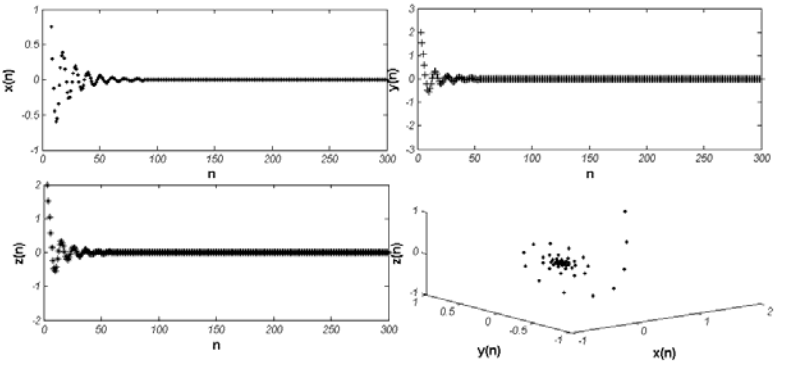


Fig. 1. The origin of (1.1) is stable when $k = 2, a = 1, \alpha = 0.01, \beta = -0.5$

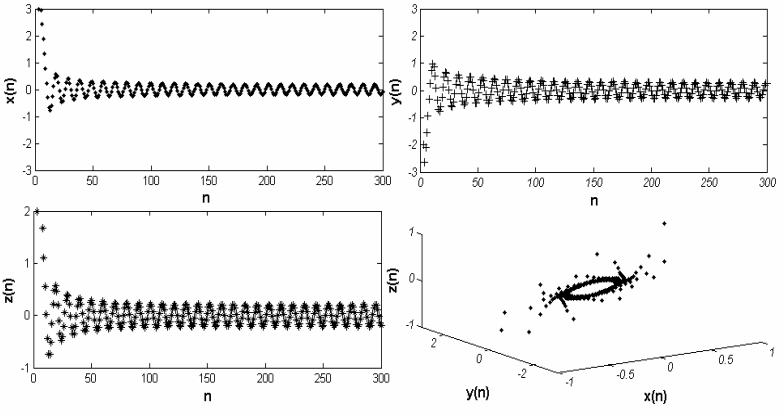


Fig. 2. A periodic solution of (1.1) is orbitally asymptotically stable at $a_{00} = -0.621$ when $k = 2$, $a = 1.02$, $\alpha = 0.022$, $\beta = -0.599$

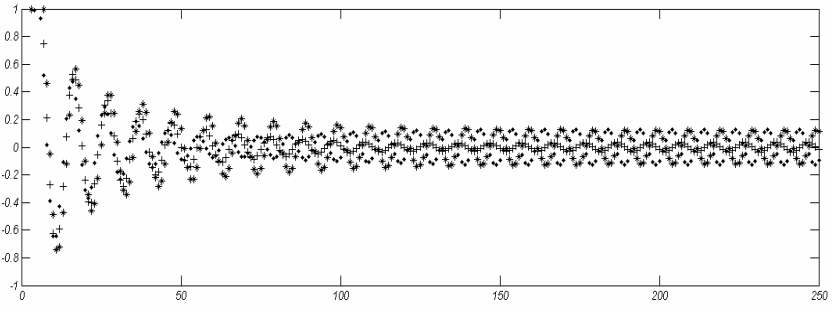


Fig. 3. Multiple periodic solution of (1.1) occurs at $a_{00} = -0.62$ when $k = 2, a = 1.0, \alpha = 0.02, \beta = -0.6$

References

1. Guo, S., Tang, X., Li, H.: Stability and bifurcation in a discrete system of two neurons with delays. Nonlinear Analysis: Real World Applications 9, 1323–1335 (2008)
2. Zhang, C., Zheng, B.: Stability and bifurcation of a two-dimension discrete neuralnetwork model with multi-delays. Chaos, Solitons and Fractals 31, 1232–1242 (2007)
3. Wei, J., Zhang, C.: Bifurcation analysis of a class of neural networks with delays. Nonlinear Analysis: Real World Applications 9, 2234–2252 (2008)
4. Zhang, C., Zheng, B.: Hopf bifurcation in numerical approximation of a n-dimension neural network model with multi-delays. Chaos, Solitons and Fractals 25, 129–146 (2005)
5. Wei, J., Ruan, S.: Stability and bifurcation in a neural network model with two delays. Physical D 130, 255–272 (1999)
6. Zhang, C., Liu, M., Zheng, B.: Hopf bifurcation in numerical pproximation of a class delay differential equations. Applied Mathematics and Computation 146, 335–349 (2003)

7. Wei, J., Zhang, C.: Stability analysis in a first-order complex differential equations with delay. Nonlinear Analysis 59, 657–671 (2004)
8. Zhao, H., Wang, L., Ma, C.: Hopf bifurcation and stability analysis on discrete-time Hopfield neural network with delay. Nonlinear Analysis: Real World Application 9, 103–113 (2008)
9. Zheng, B., Liang, L., Zhang, C.: Extended Jury Criterion. Science in China Series A: Mathematics 39, 1239–1260 (2009)
10. Yuri, A.K.: Elements of Applied Bifurcation Theory. Springer, New York (1995)
11. Golubitsky, M., Stewart, I.N., Schaeffer, D.G.: Singularities and Groups in Bifurcation Theory. Appl. Math. Sci. 2, 69 (1988)

Globally Exponential Stability of a Class of Neural Networks with Impulses and Variable Delays*

Jianfu Yang, Hongying Sun, Fengjian Yang, Wei Li, and Dongqing Wu

Department of Computation Science,
Zhongkai University of Agriculture and Engineering,
Guangzhou, 510225, P.R. China
shy_1124@sina.com, yjf197202@163.com

Abstract. In this paper, a class of impulsive neural networks with time-varying delays is considered to study the globally exponential stability. New sufficient conditions for globally exponential stability are obtained by using the vector Lyapunov function, Young inequality and Halanay differential inequality with delay.

Keywords: neural networks, globally exponential stability, Lyapunov function.

1 Introduction

Recently, the dynamical behaviors of neural networks have aroused extensive researches by numerous scholars due to their applicability in solving image processing, signal processing and pattern recognition problems. In 1988, Chua and Yang [1] put forward the cellular neural networks, Zhang Li-juan [2] studied the globally exponential stability of a class of neural networks with time-varying delays described by the system of nonlinear delay differential equations:

$$\dot{x}_i(t)=-c_i x_i(t)+\sum_{j=1}^{n} a_{ij} f_j(x_j(t))+\sum_{j=1}^{n} b_{ij} g_j(x_j(t-\tau_{ij}(t)))+J_i, t\geq 0, (i=1,2,\cdots,n) \tag{1}$$

Most widely used neural networks is neither purely continuous-time nor purely discrete-time ones, these are called impulsive neural networks [3-9]. In this paper, we consider the following impulsive neural networks:

$$\begin{cases} \dot{x}_i(t)=-c_i x_i(t)+\sum_{j=1}^{n} a_{ij} f_j(x_j(t))+\sum_{j=1}^{n} b_{ij} g_j(x_j(t-\tau_{ij}(t)))+J_i, t\neq t_k, t\geq 0, \\ \Delta x_i(t_k)=x_i(t_k^+)-x_i(t_k^-)=I_{ik}(x(t_k^-)), i=1,2,\cdots,n, k\in N\triangleq\{1,2,\cdots\}. \end{cases} \tag{2}$$

where $x(t)=(x_1(t),\cdots,x_n(t))^T$, $x_i(t)$ corresponds to the state of the ith neuron at time t, n corresponds to the number of units in the neural networks, $c>0$ denotes the neuron firing rate, a_{ij} and b_{ij} represent the connection weight and the delayed connection

* This work is supported by Science Foundation of Zhongkai University of Agriculture and Engineering G3071727.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 711–718, 2010.

weight, respectively, J_i is an external input on the ith neuron at time t,$\tau_{ij}(t)\geq 0$ is the transmission delay, and f_j, g_j are the activation functions in system (1). t_k is called impulsive moment and $t_1< t_2<\cdots<t_k<\cdots$ is a strictly increasing sequence such that $\lim_{k\to\infty} t_k=+\infty$, $x_i(t_k^-)$ and $x_i(t_k^+)$ denote the left-hand and right-hand limit at t_k, respectively, I_{ik} shows impulsive perturbation of the ith neuron at time t_k. We always assume $x_i(t_k)=x_i(t_k^-), k\in N$.

2 Preliminaries

In this section, we will study the existence and globally exponential stability of the equilibrium point of Eq. (2). To obtain our results, we need to introduce the following definitions and lemmas. Throughout this paper, we further assume that the activations f_j, g_j and the delays $\tau_{ij}(t)$ satisfy the following conditions:

H1) Each f_j and g_j are bounded continuous and for each $j=1,2,\cdots,n$, there exist real numbers $p_j>0$ and $q_j>0$ such that

$$p_j=\sup_{x\neq y}\left|(f_j(x)-f_j(y))/(x-y)\right|,\ q_j=\sup_{x\neq y}\left|(g_j(x)-g_j(y))/(x-y)\right|,$$

for every $x\neq y$.

H2) The time delays $\tau_{ij}(t)(i, j=1,2,\cdots,n)$ are bounded and nonnegative functions. That is, there is a positive number $\tau>0$ such that $0\leq\tau_{ij}(t)\leq\tau$.

The system (2) is supplemented with initial values given by

$$x_i(t_0+t)=\varphi_i(t), t\in[-\tau,0], i=1,2,\cdots,n, \tag{3}$$

$\varphi_i(t)$ denote real-valued bounded and continuous functions defined on $[-\tau,0]$. If $I_{ik}(x)=0$ for all $x\in R^n, i=1,\cdots,n, k\in N$, then the impulsive model (2) becomes continuous cellular neural networks with delays model (1).

Definition 1. (Cao and Wang [3]) A map: $H:R^n\to R^n$ is a homeomorphism of R^n onto itself, if $H\in C^0$ is onto and the inverse map $H^{-1}\in C^0$.

Definition 2. Vector function $x(t)=(x_1(t),\cdots,x_n(t))^T$ is called a solution of system (2), if $x(t)$ is continuous at $t\neq t_k$ and $t\geq 0, x(t_k)=x(t_k^-)$ and $x(t_k^+)$ exist, $x(t)$ satisfies Eq. (2) and initial condition (3). Especially, a point $x^*=(x_1^*,x_2^*,\cdots,x_n^*)$ in R^n is called an equilibrium point of (2), if $x(t)=x^*$ is a solution of (2). Where the components x_i^* are governed by the algebraic system

$$0=-c_i x_i^*+\sum_{j=1}^{n} a_{ij} f_j(x_j^*)+\sum_{j=1}^{n} b_{ij} g_j(x_j^*)+J_i \tag{4}$$

Throughout this paper, we always assume that the impulsive jumps I_k satisfy $I_k(x_i^*)=0, i=1,2,\cdots,n, k\in N$, here $x^*=(x_1^*,x_2^*,\cdots,x_n^*)^T$ is the equilibrium point of system (2). Let $x(t,t_0,\varphi)$ denote the solution of system (2) through (t_0,φ).

In the following theorem and definition, we use a result due to Forti and Tesi [10] in order to prove there is a unique solution of the system (4) under the general Euclidean norm $\|u\|_r=(\sum_{i=1}^{n}|u_i|^r)^{\frac{1}{r}}$, where $r\geq 1$ and $u=(u_1,u_2,\cdots,u_n)^T\in R^n$.

Definition 3. The equilibrium point $x^*=(x_1^*,x_2^*,\cdots,x_n^*)$ of system (2) is said to be globally exponential stable if there exist constants $\beta\geq 1$ and $\sigma>0$ such that $\|x(t)-x^*\|_r\leq\beta\|\phi-x^*\|_r e^{-\sigma t}$ for all $t\geq 0$, where

$$\|\phi-x^*\|_r=[\sup_{s\in[-\tau,0]}[\sum_{j=1}^{n}|\phi_j(s)-x_j^*|^r]]^{\frac{1}{r}}.$$

Lemma 1. (Forti and Tesi [11]) If $H(x)\in C^0$ satisfies the following conditions:

(i) $H(x)$ is injective on R^n,

(ii) $\|H(u)\|\to\infty$ as $\|u\|\to\infty$.

Then $H(x)$ is a homeomorphism of R^n onto itself.

Lemma 2. (Young inequality[12]) Assume that $\xi_1>0,\xi_2>0,l>1$ and $\frac{1}{l}+\frac{1}{m}=1$, then the inequality $\xi_1\xi_2\leq\frac{1}{l}\xi_1^l+\frac{1}{m}\xi_2^m$ holds.

Lemma 3. (Halanay inequality [13]) Let α and β be constants with $\alpha>\beta>0$ and $x(t)$ be a nonnegative continuous function on $[t_0-\tau,t_0]$. If $x(t)$ satisfies the following inequality $x'(t)\leq-\alpha x(t)+\beta\overline{x}(t)$, where $\overline{x}(t)=\sup_{t-\tau\leq s\leq t}\{x(s)\}, \tau\geq 0$ is a constant, then for $t\geq t_0$, we have $x(t)\leq\overline{x}(t_0)e^{-r(t-t_0)}$. In which r is the unique positive solution of the equation $r=\alpha-\beta e^{r\tau}$.

3 Main Result

In this section, we shall establish one globally exponential stability criteria for impulsive system (2). We consider the impulsive network (2) in which the impulsive state displacements characterized by $I_k:R\to R$ at fixed instants of time $t=t_k, k\in N$ are defined by

$$x_i(t_k)-x_i(t_k^-)=I_k(x(t_k^-))=\gamma_{ik}(x(t_k^-)-x_i^*), k\in N, i=1,2,\cdots,n. \tag{5}$$

Where γ_{ik} denote real numbers. For convenience in our analysis, we let

$$y_i(t)=x_i(t)-x_i^*, f_j(y_j(t))=f_j(y_j(t)+x_j^*)-f_j(x_j^*), g_j(y_j(t-\tau_{ij}(t)))=g_j(y_j(t-\tau_{ij}(t))+x_j^*)-g_j(x_j^*), (j=1,2,\cdots,n).$$

So that (2) can be rewritten as

$$\begin{cases} \dot{y}_i(t)=-c_i y_i(t)+\sum_{j=1}^{n} a_{ij} f_j(y_j(t))+\sum_{j=1}^{n} b_{ij} g_j(y_j(t-\tau_{ij}(t))),\ t\geq 0, \\ y_i(t_k)=(1+\gamma_{ik}) y_i(t_k^-),\ i=1,2,\cdots,n;\ k\in N, \\ y_i(s)=\psi_i(s)=\phi_i(s)-x_i^*,\ s\in[-\tau,0]. \end{cases} \tag{6}$$

Theorem 1. Let $r>1, c_i>0, a_{ij}, b_{ij}, J_i\in R$, the function $f_j(\bullet)$ and $g_j(\bullet)$ satisfy H1), and the delay $\tau_{ij}(t)$ satisfy H2), suppose the condition

H3) If there exist constants $\alpha_i>0$ such that

$$k_1=\min_{1\leq i\leq n}\{c_i-\frac{1}{r}(\sum_{j=1}^{n}\frac{\alpha_j}{\alpha_i}|a_{ji}|^{(1-\delta_{ji})r} p_i^{(1-\sigma_{ji})r}+(r-1)(|a_{ij}|^{\frac{\delta_{ij}r}{r-1}} p_j^{\frac{\sigma_{ij}r}{r-1}}+|b_{ij}|^{\frac{\delta_{ij}r}{r-1}} q_j^{\frac{\sigma_{ij}r}{r-1}}))\}>\max_{1\leq i\leq n}\{\frac{1}{r}\sum_{j=1}^{n}\frac{\alpha_j}{\alpha_i}|b_{ji}|^{(1-\delta_{ji})r} q_i^{(1-\sigma_{ji})r}\}=k_2 \tag{7}$$

is satisfied, where δ_{ij}, σ_{ij} denote real numbers. Then there exists a unique equilibrium point x^* of (2).

Proof. Consider a map $H(u)=(H_1(u), H_2(u),\cdots, H_n(u))^T\in C^0(R^n, R^n)$ endowed with the norm $\|u\|_r=(\sum_{i=1}^{n}|u_i(t)|^r)^{\frac{1}{r}}$ for $r>1$, where

$$H_i(u)=-c_i u_i+\sum_{j=1}^{n} a_{ij} f_j(u_j)+\sum_{j=1}^{n} b_{ij} g_j(u_j)+J_i, (i=1,2,\cdots,n)\cdot$$

The map $H\in C^0$ is a homeomorphism on R^n if it is injective on R^n and satisfies $\|H(u)\|_r\to\infty$ as $\|u\|_r\to\infty$.

First, we claim that H is injective on R^n, namely $H(u)=H(v)$ implies $u=v$ for any $u, v\in R^n$, as follows, we have

$$-c_i u_i+\sum_{j=1}^{n} a_{ij} f_j(u_j)+\sum_{j=1}^{n} b_{ij} g_j(u_j)+J_i=-c_i v_i+\sum_{j=1}^{n} a_{ij} f_j(v_j)+\sum_{j=1}^{n} b_{ij} g_j(v_j)+J_i,$$

and consequently

$$c_i|u_i-v_i|\leq\sum_{j=1}^{n}|a_{ij}|p_j|u_j-v_j|+\sum_{j=1}^{n}|b_{ij}|q_j|u_j-v_j|,\ i=1,2,\cdots,n,$$

under the hypotheses. Further estimation through the use of the Young inequality leads to

$$\begin{aligned}
&\sum_{i=1}^{n}\alpha_i c_i|u_i-v_i|^r\leq\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}|a_{ij}|p_j|u_j-v_j||u_i-v_i|^{r-1}+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}|b_{ij}|q_j|u_j-v_j||u_i-v_i|^{r-1} \\
&=\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}[(|a_{ij}|^{\delta_{ij}} p_j^{\sigma_{ij}}|u_i-v_i|^{r-1})\times(|a_{ij}|^{1-\delta_{ij}} p_j^{1-\sigma_{ij}}|u_j-v_j|)]+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}[(|b_{ij}|^{\delta_{ij}} q_j^{\sigma_{ij}}|u_i-v_i|^{r-1})\times(|b_{ij}|^{1-\delta_{ij}} q_j^{1-\sigma_{ij}}|u_j-v_j|)] \\
&\leq\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}[\frac{r-1}{r}|a_{ij}|^{\frac{\delta_{ij}r}{r-1}} p_j^{\frac{\sigma_{ij}r}{r-1}}|u_i-v_i|^r+\frac{1}{r}|a_{ij}|^{(1-\delta_{ij})r} p_j^{(1-\sigma_{ij})r}|u_j-v_j|^r]+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}[\frac{r-1}{r}|b_{ij}|^{\frac{\delta_{ij}r}{r-1}} q_j^{\frac{\sigma_{ij}r}{r-1}}|u_i-v_i|^r+\frac{1}{r}|b_{ij}|^{(1-\delta_{ij})r} q_j^{(1-\sigma_{ij})r}|u_j-v_j|^r] \\
&=\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}[\frac{r-1}{r}|a_{ij}|^{\frac{\delta_{ij}r}{r-1}} p_j^{\frac{\sigma_{ij}r}{r-1}}+\frac{1}{r}\frac{\alpha_j}{\alpha_i}|a_{ji}|^{(1-\delta_{ji})r} p_i^{(1-\sigma_{ji})r}]|u_i-v_i|^r+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}[\frac{r-1}{r}|b_{ij}|^{\frac{\delta_{ij}r}{r-1}} q_j^{\frac{\sigma_{ij}r}{r-1}}+\frac{1}{r}\frac{\alpha_j}{\alpha_i}|b_{ji}|^{(1-\delta_{ji})r} q_i^{(1-\sigma_{ji})r}]|u_i-v_i|^r \\
&=\sum_{i=1}^{n}\alpha_i\{\frac{1}{r}[\sum_{j=1}^{n}\frac{\alpha_j}{\alpha_i}|a_{ji}|^{(1-\delta_{ji})r} p_i^{(1-\sigma_{ji})r}+(r-1)\sum_{j=1}^{n}(|a_{ij}|^{\frac{\delta_{ij}r}{r-1}} p_j^{\frac{\sigma_{ij}r}{r-1}}+|b_{ij}|^{\frac{\delta_{ij}r}{r-1}} q_j^{\frac{\sigma_{ij}r}{r-1}})]\}|u_i-v_i|^r+\sum_{i=1}^{n}\alpha_i\{\frac{1}{r}\sum_{j=1}^{n}\frac{\alpha_j}{\alpha_i}|b_{ji}|^{(1-\delta_{ji})r} q_i^{(1-\sigma_{ji})r}\}|u_i-v_i|^r
\end{aligned}$$

which gives

$$(k_1-k_2)\sum_{i=1}^{n}\alpha_i|u_i-v_i|^r\le 0\cdot \tag{8}$$

It follows, therefore, from (8) that $u_i=v_i$ for $i=1,2,\cdots,n$ (i.e. $u=v$). Hence, the map $H\in C^0$ is injective on R^n. Second, we prove that $\|H(u)\|_r\to\infty$ as $\|u\|_r\to\infty$, we consider the map $\hat{H}(u)=H(u)-H(0)$, i.e.

$$\hat{H}_i(u)=H_i(u)-H_i(0)=-c_iu_i+\sum_{j=1}^{n}a_{ij}f_j(u_j)+\sum_{j=1}^{n}b_{ij}g_j(u_j),\ \text{for } u_i\in R,\ i=1,2,\cdots,n\ .$$

It is enough to show $\|\hat{H}(u)\|_r\to\infty$ as $\|u\|_r\to\infty$, we have

$$\begin{aligned}
&\sum_{i=1}^{n}\alpha_i|u_i|^{r-1}\mathrm{sgn}(u_i)\hat{H}_i(u)=\sum_{i=1}^{n}\alpha_i|u_i|^{r-1}\mathrm{sgn}(u_i)[-c_iu_i+\sum_{j=1}^{n}a_{ij}f_j(u_j)+\sum_{j=1}^{n}b_{ij}g_j(u_j)]\\
&\le\sum_{i=1}^{n}[-\alpha_ic_i|u_i|^r+\sum_{j=1}^{n}\alpha_i|a_{ij}|p_j|u_j||u_i|^{r-1}+\sum_{j=1}^{n}\alpha_i|b_{ij}|q_j|u_j||u_i|^{r-1}]\\
&\le\sum_{i=1}^{n}(-c_i\alpha_i|u_i|^r)+\sum_{i=1}^{n}\alpha_i\{\sum_{j=1}^{n}[\frac{r-1}{r}|a_{ij}|^{\frac{\delta_{ij}r}{r-1}}p_j^{\frac{\sigma_{ij}r}{r-1}}|u_i|^r+\frac{1}{r}|a_{ij}|^{(1-\delta_{ij})r}p_j^{(1-\sigma_{ij})r}|u_j|^r]+\sum_{j=1}^{n}[\frac{1}{r}|b_{ij}|^{(1-\delta_{ij})r}q_j^{(1-\sigma_{ij})r}|u_j|^r+\frac{r-1}{r}|b_{ij}|^{\frac{\delta_{ij}r}{r-1}}q_j^{\frac{\sigma_{ij}r}{r-1}}|u_i|^r]\}\\
&\le\sum_{i=1}^{n}[-c_i\alpha_i|u_i|^r]+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}\frac{1}{r}[(r-1)(|a_{ij}|^{\frac{\delta_{ij}r}{r-1}}p_j^{\frac{\sigma_{ij}r}{r-1}}+|b_{ij}|^{\frac{\delta_{ij}r}{r-1}}q_j^{\frac{\sigma_{ij}r}{r-1}})+\frac{\alpha_j}{\alpha_i}|a_{ji}|^{(1-\delta_{ji})r}p_i^{(1-\sigma_{ji})r}]|u_i|^r+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}[\frac{1}{r}\frac{\alpha_j}{\alpha_i}|b_{ji}|^{(1-\delta_{ji})r}q_i^{(1-\sigma_{ji})r}]|u_i|^r\\
&=-\sum_{i=1}^{n}\alpha_i\{c_i-\sum_{j=1}^{n}\frac{1}{r}[(r-1)(|a_{ij}|^{\frac{\delta_{ij}r}{r-1}}p_j^{\frac{\sigma_{ij}r}{r-1}}+|b_{ij}|^{\frac{\delta_{ij}r}{r-1}}q_j^{\frac{\sigma_{ij}r}{r-1}})+\frac{\alpha_j}{\alpha_i}|a_{ji}|^{(1-\delta_{ji})r}p_i^{(1-\sigma_{ji})r}]-\sum_{j=1}^{n}\frac{1}{r}[\frac{\alpha_j}{\alpha_i}|b_{ji}|^{(1-\delta_{ji})r}q_i^{(1-\sigma_{ji})r}]\}|u_i|^r\\
&\le-(k_1-k_2)\sum_{i=1}^{n}\alpha_i|u_i|^r,
\end{aligned}$$

that yields

$$(k_1-k_2)\sum_{i=1}^{n}\alpha_i|u_i|^r\le-\sum_{i=1}^{n}\alpha_i|u_i|^{r-1}\mathrm{sgn}(u_i)\hat{H}_i(u)\le\sum_{i=1}^{n}\alpha_i|u_i|^{r-1}|\hat{H}_i(u)|$$

On applying a Hölder inequality

$$(k_1-k_2)\min_{i=1,\cdots,n}\{\alpha_i\}\sum_{i=1}^{n}\alpha_i|u_i|^r\le\max_{i=1,\cdots,n}\{\alpha_i\}(\sum_{i=1}^{n}|u_i|^{(r-1)s})^{\frac{1}{s}}(\sum_{i=1}^{n}|\hat{H}_i(u)|^r)^{\frac{1}{r}},$$

where $\frac{1}{r}+\frac{1}{s}=1$. It then follows that

$$(\sum_{i=1}^{n}|u_i|^r)^{\frac{1}{r}}\le\frac{\max_{i=1,\cdots,n}\{\alpha_i\}}{(k_1-k_2)\min_{i=1,\cdots,n}\{\alpha_i\}}(\sum_{i=1}^{n}|\hat{H}_i(u)|^r)^{\frac{1}{r}}$$

and from which we assert $\|H(u)\|_r\to\infty$ as $\|u\|_r\to\infty$. We conclude that the map $H\in C^0$ is a homeomorphism on R^n and this guarantees the existence of a unique solution $x^*\in R^n$ of the algebraic system (4) which defines the unique equilibrium state of the impulsive network (2). The proof is now complete.

Theorem 2. Let $r>1, c_i>0, a_{ij}, b_{ij}, J_i\in R$, in addition to H1), H2), H3) and (5) hold, assume further that:

H4) There exists a constant γ such that

$$\frac{\ln\gamma_k}{t_k-t_{k-1}}\le\gamma<\lambda,\ k=1,2,\cdots,$$

where $\gamma_k=\max\{1,|1+\gamma_{1k}|^r,\cdots,|1+\gamma_{nk}|^r\},k\in N$, the scalar $\lambda>0$ is the unique positive solution of the equation:

$$r\lambda=rk_1-rk_2e^{r\lambda t}.$$

Then the equilibrium point x^* of system (2) is globally exponential stable.

Proof. According to assumption H1), we get

$$\left|f_j(y_j(t)+x_j^*)-f_j(x_j^*)\right|\le p_j\left|y_j(t)\right|,\ \left|g_j(y_j(t-\tau_{ij}(t))+x_j^*)-g_j(x_j^*)\right|\le q_j\left|y_j(t-\tau_{ij}(t))\right|. \tag{9}$$

We consider the Lyapunov function $V(t)=\sum_{i=1}^{n}\alpha_i|y_i(t)|^r$, we denote $\bar{V}(t)=\sup_{t-\tau\le s\le t}V(s)=\sup_{t-\tau\le s\le t}(\sum_{i=1}^{n}\alpha_i|y_i(s)|^r)$ for $t\ge0$. From (8), the upper right Dini derivative $D^+V(t)$ of $V(t)$ along the solutions of system (6) is obtained as

$$\begin{aligned}
&D^+V(t)\\
&=\sum_{i=1}^{n}r\alpha_i|y_i(t)|^{r-1}\operatorname{sgn}(y_i(t))[-c_iy_i(t)+\sum_{j=1}^{n}a_{ij}(f_j(y_j(t)+x_j^*)-f_j(x_j^*))+\sum_{j=1}^{n}b_{ij}(g_j(y_j(t-\tau_{ij}(t))+x_j^*)-g_j(x_j^*))]\\
&\le\sum_{i=1}^{n}r\alpha_i[-c_i|y_i(t)|^r+\sum_{j=1}^{n}p_j|a_{ij}||y_j(t)||y_i(t)|^{r-1}+\sum_{j=1}^{n}q_j|b_{ij}||y_j(t-\tau_{ij}(t))||y_i(t)|^{r-1}]\\
&=\sum_{i=1}^{n}r\alpha_i[-c_i|y_i(t)|^r+\sum_{j=1}^{n}[(|a_{ij}|^{\delta_{ij}}p_j^{\sigma_{ij}}|y_i(t)|^{r-1})\times(|a_{ij}|^{1-\delta_{ij}}p_j^{1-\sigma_{ij}}|y_j(t)|)]+\sum_{j=1}^{n}[(|b_{ij}|^{\delta_{ij}}q_j^{\sigma_{ij}}|y_i(t)|^{r-1})\times(|b_{ij}|^{1-\delta_{ij}}q_j^{1-\sigma_{ij}}|y_j(t-\tau_{ij}(t))|)]]\\
&\le\sum_{i=1}^{n}r\alpha_i[-c_i|y_i(t)|^r+\frac{1}{r}\sum_{j=1}^{n}|a_{ij}|^{(1-\delta_{ij})r}p_j^{(1-\sigma_{ij})r}|y_j(t)|^r+\frac{r-1}{r}\sum_{j=1}^{n}|a_{ij}|^{\frac{\delta_{ij}r}{r-1}}p_j^{\frac{\sigma_{ij}r}{r-1}}|y_i(t)|^r+\frac{r-1}{r}\sum_{j=1}^{n}|b_{ij}|^{\frac{\delta_{ij}r}{r-1}}q_j^{\frac{\sigma_{ij}r}{r-1}}|y_i(t)|^r\\
&\quad+\frac{1}{r}\sum_{j=1}^{n}|b_{ij}|^{(1-\delta_{ij})r}q_j^{(1-\sigma_{ij})r}|y_j(t-\tau_{ij}(t))|^r]\\
&=\sum_{i=1}^{n}r\alpha_i[-c_i+\frac{1}{r}\sum_{j=1}^{n}\frac{\alpha_j}{\alpha_i}|a_{ji}|^{(1-\delta_{ji})r}p_i^{(1-\sigma_{ji})r}+\frac{r-1}{r}\sum_{j=1}^{n}(|a_{ij}|^{\frac{\delta_{ij}r}{r-1}}p_j^{\frac{\sigma_{ij}r}{r-1}}+|b_{ij}|^{\frac{\delta_{ij}r}{r-1}}q_j^{\frac{\sigma_{ij}r}{r-1}})]|y_i(t)|^r\\
&\quad+\sum_{i=1}^{n}\alpha_i\sum_{j=1}^{n}\frac{\alpha_j}{\alpha_i}|b_{ji}|^{(1-\delta_{ji})r}q_i^{(1-\sigma_{ji})r}|y_i(t-\tau_{ji}(t))|^r\\
&\le-r\min_{1\le i\le n}\{c_i-\frac{1}{r}\sum_{j=1}^{n}((r-1)(|a_{ij}|^{\frac{\delta_{ij}r}{r-1}}p_j^{\frac{\sigma_{ij}r}{r-1}}+|b_{ij}|^{\frac{\delta_{ij}r}{r-1}}q_j^{\frac{\sigma_{ij}r}{r-1}})+\frac{\alpha_j}{\alpha_i}|a_{ji}|^{(1-\delta_{ji})r}p_i^{(1-\sigma_{ji})r})\}V(t)+r\max_{1\le i\le n}\{\frac{1}{r}\sum_{j=1}^{n}\frac{\alpha_j}{\alpha_i}|b_{ji}|^{(1-\delta_{ji})r}q_i^{(1-\sigma_{ji})r}\}\bar{V}(t)\\
&=-rk_1V(t)+rk_2\bar{V}(t).
\end{aligned}$$

According to assumption H3) and Lemma 3, it is obviously that

$$V(t)\le\bar{V}(t_0)e^{-r\lambda(t-t_0)}=\bar{V}(0)e^{-r\lambda t}$$

for $0=t_0<t<t_1$. Where λ is the unique positive solution of the equation $r\lambda=rk_1-rk_2e^{r\lambda t}$. Also, in view of condition (5), one has

$$x_i(t_1)-x_i^*=x_i(t_1^-)-x_i^*+I_{i1}(x_i(t_1^-)-x_i^*)=(1+\gamma_{i1})(x_i(t_1^-)-x_i^*)\ .$$

Then at $t=t_1$,

$$V(t_1^+)=\sum_{i=1}^{n}\alpha_i|y_i(t_1^+)|^r=\sum_{i=1}^{n}\alpha_i|1+\gamma_{i1}|^r|y_i(t_1^-)|^r\le|\gamma_1|^r\sum_{i=1}^{n}\alpha_i|y_i(t_1^-)|^r=\gamma_1^r V(t_1^-)\le\gamma_1^r V(0).$$

By following the similar inductive arguments as before, we derive that

$$V(t_{k-1}^+)\le\overline{V}(0)\gamma_0^r\gamma_1^r\gamma_2^r\cdots\gamma_{k-1}^r e^{-r\lambda t_{k-1}}$$

for $t=t_{k-1},k\in N$, where $t_0=0$ and $\gamma_0^r=1$, by the mathematical induction, we can conclude that

$$V(t)\le\overline{V}(0)\gamma_0^r\gamma_1^r\gamma_2^r\cdots\gamma_{k-1}^r e^{-r\lambda t}. \tag{10}$$

For $t_{k-1}<t<t_k,k\in N$, noticing that $\gamma_k\le e^{\gamma(t_k-t_{k-1})}$ by H4), we can use (10) to conclude that

$$V(t)\le\overline{V}(0)e^{r\gamma(t_1-t_0)}e^{r\gamma(t_2-t_1)}\cdots e^{r\gamma(t_k-t_{k-1})}e^{-r\lambda t}\le\overline{V}(0)e^{r\gamma(t-t_0)}e^{-r\lambda t}=\overline{V}(0)e^{r\gamma t}e^{-r\lambda t}=\overline{V}(0)e^{-r(\lambda-\gamma)t}$$

for $0<t<t_k, k\in N$, which is equivalent to

$$\sum_{i=1}^{n}|y_i(t)|^r\le\beta(\|\psi\|_r)^r e^{-r(\lambda-\gamma)t}$$

for $0<t<t_k, k\in N$, where

$$\beta=\max_{1\le i\le n}\{\alpha_i\}/\min_{1\le i\le n}\{\alpha_i\}\ge 1,\ \|\psi\|_r=\|\phi-x^*\|_r=\{\sup_{s\in[-\tau,0]}[\sum_{j=1}^{n}|\phi_j(s)-x_j^*|^r]\}^{\frac{1}{r}}.$$

Hence

$$[\sum_{i=1}^{n}|y_i(t)|^r]^{\frac{1}{r}}\le\sqrt[r]{\beta}\|\psi\|_r e^{-(\lambda-\gamma)t}\ \text{ for } 0<t<t_k,k\in N.$$

So the vector solution $y(t)$ converges to the equilibrium state $y^*=0$ of the impulsive network (6) exponentially in the norm $\|u\|_r=(\sum_{i=1}^{n}|u_i|^r)^{\frac{1}{r}}$ as $t\to\infty$, that is, the equilibrium point x^* of system (2) is globally exponentially stable. This completes the proof.

4 Conclusion

In this paper, we consider the globally exponential stability of the neural networks with variable delays and impulses. Under assumption that the activation function is not bounded, monotonic or differentiable, we obtain some sufficient conditions for the existence and globally exponential stability of a unique equilibrium.

References

1. Chua, L.O., Yang, L.: Cellular neural networks: Theory and application. IEEE Transactions on Circuits and Systems 35, 1257–1272 (1988)
2. Zhang, L.-j., Si, L.-g.: Globally exponential stability of a class of neural networks with variable delays. Mathematica Applicata 20(2), 258–262 (2007)
3. Cao, J.D., Wang, J.: Global exponential stability and periodicity of recurrent neural networks with time delay. IEEE Trans. Circuits and Systems 152, 920–931 (2005)
4. Akca, H., Alassar, R., Covacheva, V., Ai-Zahrani, E.: Continuous-time additive Hopfield type neural networks with impulses. J. Math. Anal. Appl. 290, 436–451 (2004)
5. Gopalsamy, K.: Stability of artificial neural networks with impulses. Appl. Math. & Comp. 154, 783–813 (2004)
6. Xia, Y.H., Cao, J.D., Cheng, S.: Global exponential stability of delayed cellular neural networks with impulses. Neuro-computing 70, 2495–2501 (2007)
7. Mohamad, S., Gopalsamy, K., Akca, H.: Exponential stability of artificial neural networks with distributed delays and large impulses. Nonlinear Analysis: Real World Applications 9, 872–888 (2008)
8. Yang, Z.C., Xu, D.Y.: Impulsive effects on stability of Cohen-Grossberg neural networks with variable delays. Applied Math. And Comput. 160, 1–16 (2005)
9. Yang, Y., Cao, J.D.: Stability and periodicity in delayed cellular neural networks with impulsive effects Nonlinear Analysis. Real World Applications 8, 362–374 (2007)
10. Yang, T.: Impulsive systems and control: Theory and Applications. Nova science publishers, Huntington (2001)
11. Forti, M., Tesi, A.: New conditions for global stability of neural networks with application to linear and quadratic programming problems. IEEE Trans. Circuits Syst. I: Fund. Theor. Appl. 42, 354–366 (1995)
12. Hardy, G.H., Littlewood, J.E., Polya, G.: Inequalities. Cambridge University Press, London (1952)
13. Gopalsamy, K.: Stability and Oscillations in Delay Differential Equations of Population Dynamics. Kluwer Academic Publishers, Dordrecht (1992)

Discrete Time Nonlinear Identification via Recurrent High Order Neural Networks for a Three Phase Induction Motor

Alma Y. Alanis[1], Edgar N. Sanchez[2],
Alexander G. Loukianov[2], and Marco A. Perez-Cisneros[1]

[1] Departamento de Ciencias Computacionales, CUCEI, Universidad de Guadalajara, Apartado Postal 51-71, Col. las Aguilas, Zapopan, Jalisco, Mexico, C.P. 45079
almayalanis@gmail.com
[2] CINVESTAV, Unidad Guadalajara, Apartado Postal 31-438, Plaza La Luna, Guadalajara, Jalisco, C.P. 45091, Mexico
sanchez@gdl.cinvestav.mx

Abstract. This paper deals with the problem of discrete-time nonlinear system identification via Recurrent High Order Neural Networks. It includes the respective stability analysis on the basis of the Lyapunov approach for the extended Kalman filter (EKF)-based NN training algorithm, which is applied for learning. Applicability of the scheme is illustrated via simulation for a discrete-time nonlinear model of an electric induction motor.

1 Introduction

Neural networks (NN) have become a well-established methodology as exemplified by their applications to identification and control of general nonlinear and complex systems. In particular, the use of recurrent high order neural networks (RHONN) has increased recently [8]. There are recent results which illustrate that the NN technique is highly effective in the identification of a broad category of complex discrete-time nonlinear systems without requiring complete model information ([12], [13]).

Lyapunov approach can be used directly to obtain robust training algorithms for countinuous-time recurrent neural networks ([8], [9]). For discrete-time systems, the problem is more complex due to the couplings among subsystems, inputs and outputs. Few results have been published in comparison with those for continuous-time domain ([12], [13]). By other hand discrete-time neural networks are more convenient for real-time applications.

For many nonlinear systems it is often difficult to obtain their accurate and faithful mathematical models, regarding their physically complex structures and hidden parameters as discussed in [2]. Therefore, system identification becomes important and even necessary before system control can be considered not only for understanding and predicting the behavior of the system, but also for obtaining an effective control law.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 719–726, 2010.

The identification problem consists of choosing an appropriate identification model and adjusting its parameters according to some adaptive law, such that the response of the model to an input signal (or class of input signals), approximates the response of the real system to the same input [9].

Several training methods for discrete-time recurrent networks have been proposed in the literature as a viable alternative. New training algorithms, e.g., those based on Kalman filtering, have appeared [4], [5], [10]. In this paper, we use an Extended Kalman Filter (EKF)-based training algorithm for the RHONN, in order to identify discrete-time nonlinear systems.

2 Mathematical Preliminaries

Through this paper we use k as the step sampling, $k \in 0 \cup \mathbb{Z}^+$, $|\bullet|$ for the absolute value, $\|\bullet\|$ for the Euclidian norm for vectors and for any adequate norm for matrices. For more details related to this section see [3]. Consider a MIMO nonlinear system:

$$\chi(k+1) = F(\chi(k), u(k)) \tag{1}$$

where $\chi \in \Re^n$, $u \in \Re^m$ and $F \in \Re^n \times \Re^m \to \Re^n$ is nonlinear function.

2.1 Discrete-Time Recurrent Neural Networks

Consider the following discrete-time recurrent high order neural network (RHONN):

$$x_i(k+1) = w_i^\top z_i(x(k), u(k)), \quad i = 1, \cdots, n \tag{2}$$

where x_i $(i = 1, 2, \cdots, n)$ is the state of the ith neuron, L_i is the respective number of higer-order connections, $\{I_1, I_2, \cdots, I_{L_i}\}$ is a collection of non-ordered subsets of $\{1, 2, \cdots, n\}$, n is the state dimension, w_i $(i = 1, 2, \cdots, n)$ is the respective on-line adapted weight vector, and $z_i(x(k), u(k))$ is given by

$$z_i(x(k), u(k)) = \begin{bmatrix} z_{i_1} \\ z_{i_2} \\ \vdots \\ z_{i_{L_i}} \end{bmatrix} = \begin{bmatrix} \Pi_{j \in I_1} y_{i_j}^{d_{i_j}(1)} \\ \Pi_{j \in I_2} y_{i_j}^{d_{i_j}(2)} \\ \vdots \\ \Pi_{j \in I_{L_i}} y_{i_j}^{d_{i_j}(L_i)} \end{bmatrix} \tag{3}$$

with $d_{j_i}(k)$ being a nonnegative integers, and y_i is defined as follows:

$$y_i = \begin{bmatrix} y_{i_1} \\ \vdots \\ y_{i_n} \\ y_{i_{n+1}} \\ \vdots \\ y_{i_{n+m}} \end{bmatrix} = \begin{bmatrix} S(x_1) \\ \vdots \\ S(x_n) \\ u_1 \\ \vdots \\ u_m \end{bmatrix} \tag{4}$$

In (4), $u = [u_1, u_2, \ldots, u_m]^\top$ is the input vector to the neural network, and $S(\bullet)$ is defined by

$$S(x) = \frac{1}{1 + \exp(-\beta x)} \tag{5}$$

Consider the problem to approximate the general discrete-time nonlinear system (1), by the following discrete-time RHONN serie-parallel representation [9]:

$$\chi_i(k+1) = w_i^{*\top} z_i(x(k), u(k)) + \epsilon_{z_i} \tag{6}$$

where χ_i is the ith plant state, ϵ_{z_i} is a bounded approximation error, which can be reduced by increasing the number of the adjustable weights [9]. Assume that there exists ideal weights vector w_i^* such that $\|\epsilon_{z_i}\|$ can be minimized on a compact set $\Omega_{z_i} \subset \Re^{L_i}$ The ideal weight vector w_i^* is an artificial quantity required for analytical purpose [9]. In general it is assumed that this vector exists and is constant but unknown. Let us define its estimate as w_i and the estimation error as

$$\widetilde{w}_i(k) = w_i^* - w_i(k) \tag{7}$$

The estimate w_i is used for stability analysis which will be discussed later. Since w_i^* is constant, then $\widetilde{w}_i(k+1) - \widetilde{w}_i(k) = w_i(k+1) - w_i(k)$, $\forall k \in 0 \cup \mathbb{Z}^+$.

2.2 The EKF Training Algorithm

Kalman filtering (KF) estimates the state of a linear system with additive state and output white noises [2], [4]. For KF-based neural network training, the network weights become the states to be estimated, with the error between the neural network output and the desired output being considered; this error is considered as additive white noise. For identification, the desired output is information generated by the plant; in this paper, the respective state. Due to the fact that the neural network mapping is nonlinear, an extended Kalman Filtering (EKF)-type is required.

The training goal is to find the optimal weight values that minimize the prediction errors (the differences between the desired outputs and the neural network outputs). The EKF-based NN training algorithm is described by

$$\begin{aligned} K_i(k) &= P_i(k) H_i(k) M_i(k) \qquad i = 1, \cdots, n \\ w_i(k+1) &= w_i(k) + \eta_i K_i(k) e_i(k) \\ P_i(k+1) &= P_i(k) - K_i(k) H_i^\top(k) P_i(k) + Q_i(k) \end{aligned} \tag{8}$$

with

$$M_i(k) = \left[R_i(k) + H_i^\top(k) P_i(k) H_i(k)\right]^{-1} \tag{9}$$

$$e_i(k) = \chi_i(k) - x_i(k) \tag{10}$$

where $e_i(k)$ is the respective identification error, $P_i(k) \in \Re^{L_i \times L_i}$ is the prediction error covariance matrix at step k, $w_i \in \Re^{L_i}$ is the weight (state) vector, L_i is the respective number of neural network weights, χ_i is the ith plant state, x_i is the ith neural network state, n is the number of states, $K_i \in \Re^{L_i}$ is the Kalman gain vector, $Q_i \in \Re^{L_i \times L_i}$ is the NN weight estimation noise covariance matrix, $R_i \in \Re$ is the error noise covariance; $H_i \in \Re^{L_i}$ is a vector, in which each entry (H_{ij}) is the derivative of one of the neural network state, (x_i), with respect to one neural network weight, (w_{ij}), as follows

$$H_{ij}(k) = \left[\frac{\partial x_i(k)}{\partial w_{ij}(k)}\right]^{\top}_{w_i(k)=w_i(k+1)} \tag{11}$$

where $i = 1, ..., n$ and $j = 1, ..., L_i$

Usually $P_i$and Q_i are initialized as diagonal matrices, with entries $P_i(0)$ and $Q_i(0)$, respectively. It is important to remark that $H_i(k)$, $K_i(k)$ and $P_i(k)$ for the EKF are bounded; for a detailed explanation of this fact see [11].

Then the dynamics of the identification error (10) can be expressed as

$$e_i(k+1) = \widetilde{w}_i(k) z_i(x(k), u(k)) + \epsilon_{z_i} \tag{12}$$

By the other hand the dynamics of (7) is

$$\widetilde{w}_i(k+1) = \widetilde{w}_i(k) - \eta_i K_i(k) e(k) \tag{13}$$

Now, we establish the main result of this paper in the following theorem.

Theorem 2: The RHONN (2) trained with the EKF-based algorithm (8) to identify the nonlinear plant (1), ensures that the identification error (10) is semiglobally uniformly ultimately bounded (SGUUB); moreover, the RHONN weights remain bounded.

Proof. Due to space limitations, the stability proof it is not included in this paper. Please see [1].

3 Application

In this section we apply the above developed scheme to a three-phase induction motor model

3.1 Motor Model

The six-order discrete-time induction motor model in the stator fixed reference frame (α, β) under the assumptions of equal mutual inductances and linear magnetic circuit is given by [7]

$$
\begin{aligned}
\omega(k+1) &= \omega(k) + \frac{\mu}{\alpha}(1-\alpha) M \left(i^{\beta}(k)\psi^{\alpha}(k) - i^{\alpha}(k)\psi^{\beta}(k)\right) - \left(\frac{T}{J}\right) T_L(k) \\
\psi^{\alpha}(k+1) &= \cos\left(n_p\theta(k+1)\right)\rho_1(k) - \sin\left(n_p\theta(k+1)\right)\rho_2(k) \\
\psi^{\beta}(k+1) &= \sin\left(n_p\theta(k+1)\right)\rho_1(k) + \cos\left(n_p\theta(k+1)\right)\rho_2(k) \\
i^{\alpha}(k+1) &= \varphi^{\alpha}(k) + \frac{T}{\sigma}u^{\alpha}(k) \\
i^{\beta}(k+1) &= \varphi^{\beta}(k) + \frac{T}{\sigma}u^{\beta}(k) \\
\theta(k+1) &= \theta(k) + \omega(k)T + \frac{\mu}{\alpha}\left[T - \frac{(1-a)}{\alpha}\right] \\
&\quad \times M\left(i^{\beta}(k)\psi^{\alpha}(k) - i^{\alpha}(k)\psi^{\beta}(k)\right) - \frac{T_L(k)}{J}T^2
\end{aligned}
\tag{14}
$$

with

$$
\begin{aligned}
\rho_1(k) &= a\left(\cos\left(\phi(k)\right)\psi^{\alpha}(k) + \sin\left(n_p\phi(k)\right)\psi^{\beta}(k)\right) \\
&\quad + b\left(\cos\left(\phi(k)\right)i^{\alpha}(k) + \sin\left(\phi(k)\right)i^{\beta}(k)\right) \\
\rho_2(k) &= a\left(\cos\left(\phi(k)\right)\psi^{\alpha}(k) - \sin\left(\phi(k)\right)\psi^{\beta}(k)\right) \\
&\quad + b\left(\cos\left(\phi(k)\right)i^{\alpha}(k) - \sin\left(\phi(k)\right)i^{\beta}(k)\right) \\
\varphi^{\alpha}(k) &= i^{\alpha}(k) + \alpha\beta T\psi^{\alpha}(k) + n_p\beta T\omega(k)\psi^{\alpha}(k) - \gamma T i^{\alpha}(k) \\
\varphi^{\beta}(k) &= i^{\beta}(k) + \alpha\beta T\psi^{\beta}(k) + n_p\beta T\omega(k)\psi^{\beta}(k) - \gamma T i^{\beta}(k) \\
\phi(k) &= n_p\theta(k)
\end{aligned}
\tag{15}
$$

with $b = (1-a)M$, $\alpha = \frac{R_r}{L_r}$, $\gamma = \frac{M^2R_r}{\sigma L_r^2} + \frac{R_s}{\sigma}$, $\sigma = L_s - \frac{M^2}{L_r}$, $\beta = \frac{M}{\sigma L_r}$, $a = e^{-\alpha T}$ and $\mu = \frac{Mn_p}{JL_r}$, besides Ls, L_r and M are the stator, rotor and mutual inductance respectively; R_s and R_r are the stator and rotor resistances respectively; n_p is the number of pole pairs; i^{α} and i^{β} represents the currents in the α and β phases, respectively; ψ^{α} and ψ^{β} represents the fluxes in the α and β phases, respectively and θ is the rotor angular displacement. Simulations are performed for the system (14), using the following parameters: $R_s = 14\Omega$; $L_s = 400mH$; $M = 377mH$; $R_r = 10.1\Omega$; $L_r = 412.8mH$; $n_p = 2$; $J = 0.01Kgm^2$; $T = 0.001s$.

3.2 Neural Network Identification

The RHONN proposed for this application is as follows:

$$
\begin{aligned}
x_1(k+1) &= w_{11}(k)S(\omega(k)) + w_{12}(k)S(\omega)S\left(\psi^{\beta}(k)\right)i^{\alpha}(k) \\
&\quad + w_{13}(k)S(\omega)S\left(\psi^{\alpha}(k)\right)i^{\beta}(k) \\
x_2(k+1) &= w_{21}(k)S(\omega(k))S\left(\psi^{\beta}(k)\right) + w_{22}(k)i^{\beta}(k)
\end{aligned}
$$

$$
\begin{aligned}
x_3(k+1) &= w_{31}(k)\, S(\omega(k))\, S(\psi^{\alpha}(k)) + w_{32}(k)\, i^{\alpha}(k) \\
x_4(k+1) &= w_{41}(k)\, S(\psi^{\alpha}(k)) + w_{42}(k)\, S\left(\psi^{\beta}(k)\right) \\
&\quad + w_{43}(k)\, S(i^{\alpha}(k)) + w_{44}(k)\, u^{\alpha}(k) \\
x_5(k+1) &= w_{51}(k)\, S(\psi^{\alpha}(k)) + w_{52}(k)\, S\left(\psi^{\beta}(k)\right) \\
&\quad + w_{53}(k)\, S\left(i^{\beta}(k)\right) + w_{54}(k)\, u^{\beta}(k)
\end{aligned}
$$

The training is performed on-line, using a series-parallel configuration. During the identification process the plant and the NN operates in open-loop. Both of them (plant and NN) have the same input vector $\left[\, u_{\alpha}\ u_{\beta} \,\right]^{\top}$; u_{α} and u_{β} are chirps functions with 170volts of maximal amplitude and incremental frecuencies from 0Hz to 250Hz and 0Hz to 200Hz respectively. All the NN states are initialized in a random way as well as the weights vectors. It is important remark that the initial conditions of the plant are completely different from the initial conditions for the NN. The identification is performed using (8) with $i = 1, 2, \cdots, n$ with n the dimension of plant states ($n = 6$).

3.3 Simulation Results

The results of the simulation are presented in Figs. 1-5. Fig. 1 displays the identification performance for the speed rotor; Fig. 2 and Fig. 3 present the identification performance for the fluxes in phase α and β, respectively. Figs 4 and 5 portray the identification performance for currents in phase α and β, respectively.

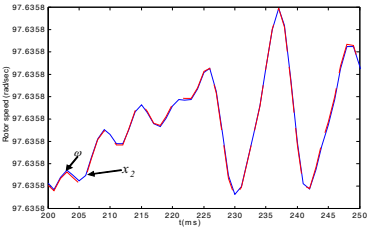


Fig. 1. Rotor speed identification

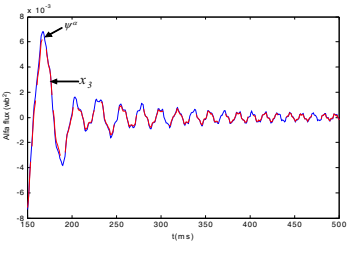


Fig. 2. ψ^{α} identification

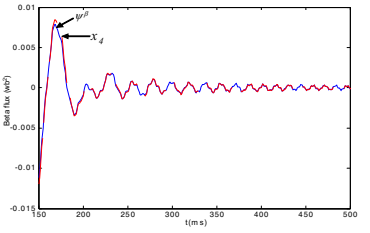


Fig. 3. ψ^β identification

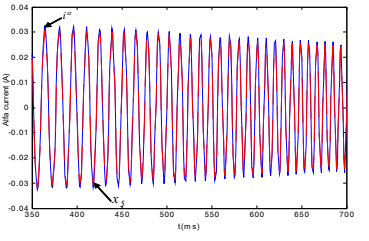


Fig. 4. i^α identification

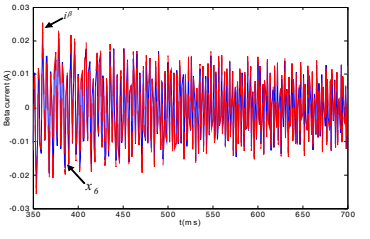


Fig. 5. i^β identification

4 Conclusions

This paper has presented the application of recurrent high order neural networks to identification of discrete-time nonlinear systems. The training of the neural networks was performed on-line using an extended Kalman filter. The boundness of the identification error was established on the basis of the Lyapunov approach. Simulation results illustrate the applicability of the proposed identification methodology. Researches are being pursued to develop new discrete-time nonlinear adaptive control based on the discussed identification scheme.

Acknowledgements. The authors thank the support of CONACYT Mexico, through Projects 57801Y and 103191Y.

References

1. Alanis, A.Y., Sanchez, E.N., Loukianov, A.G., Perez-Cisneros, M.A.: Real-time discrete neural block control using sliding modes for electric induction motors. IEEE Transactions on Control Systems Technology 18(1), 11–21 (2009)
2. Chui, C.K., Chen, G.: Kalman Filtering with Real-Time Applications, 3rd edn. Springer, New York (1998)
3. Ge, S.S., Zhang, J., Lee, T.H.: Adaptive neural network control for a class of MIMO nonlinear systems with disturbances in discrete-time. IEEE Transactions on Systems, Man and Cybernetics, Part B 34(4) (August 2004)
4. Grover, R., Hwang, P.Y.C.: Introduction to Random Signals and Applied Kalman Filtering, 2nd edn. John Wiley and Sons, New York (1992)
5. Haykin, S.: Neural Networks. A comprehensive foundation, 2nd edn. Prentice Hall, New Jersey (1999)
6. Kim, Y.H., Lewis, F.L.: High-Level Feedback Control with Neural Networks. World Scientific, Singapore (1998)
7. Loukianov, A.G., Rivera, J., Cañedo, J.M.: Discrete-time sliding mode control of an induction motor. In: Proceedings IFAC 2002, Barcelone, Spain (July 2002)
8. Sanchez, E.N., Ricalde, J.L.: Trajectory tracking via adaptive recurrent neural control with input saturation. In: Proceedings of International Joint Conference on Neural Networks 2003, Portland, Oregon, USA (July 2003)
9. Rovithakis, G.A., Chistodoulou, M.A.: Adaptive Control with Recurrent High - Order Neural Networks. Springer, New York (2000)
10. Singhal, S., Wu, L.: Training multilayer perceptrons with the extended Kalman algorithm. In: Touretzky, D.S. (ed.) Advances in Neural Information Processing Systems, vol. 1, pp. 133–140. Morgan Kaufmann, San Mateo (1989)
11. Song, Y., Grizzle, J.W.: The extended Kalman Filter as Local Asymptotic Observer for Discrete-Time Nonlinear Systems. Journal of Mathematical systems, Estimation and Control 5(1), 59–78 (1995)
12. Yu, W., Li, X.: Discrete-time neuro identification without robust modification. IEE Proc-Control Theory Appl. 150(3) (May 2003)
13. Yu, W., Li, X.: Nonlinear system identification using discrete-time recurrent neural networks with stable learning algorithms. Information Sciences 158, 131–147 (2004)

Stability Analysis for Stochastic BAM Neural Networks with Distributed Time Delays

Guanjun Wang

Department of Mathematics, Southeast University, Nanjing 210096, China
wgjmath@gmail.com

Abstract. This paper is concerned with the stability problem for stochastic bidirectional associative memory neural networks with distributed time delays. By applying the Lyapunov functional method, the semimartingale convergence theorem, and some inequality techniques, some sufficient conditions are derived to guarantee the almost sure exponential stability of the system. One example is provided at the end of this paper to show the effectiveness of our results.

Keywords: Almost surely exponentially stable, BAM neural networks, Lyapunov functional, Distributed time delays, Itô's formula.

1 Introduction

Bidirectional associative memory (BAM) neural networks was firstly proposed by Kosko in 1987 [1,2]. The dynamics analysis for BAM neural networks have attracted much attention from scholars due to their potential applications in associative memory, parallel computation and pattern recognition etc. Some results have been reported in [3,4,5].

In practice, time-delays are often encountered in various neural networks because of the finite speed of information transmission. The existence of time-delays may affect the dynamic behaviors of neural networks and therefore is of great significance to be considered in research. Discrete time delays are often used in modeling neural networks. However, in some cases, distributed time delays are more suitable in modeling the transmission processes. BAM neural networks with distributed time delays can be described by:

$$\begin{cases} \dot{u}_i(t) = -c_i u_i(t) + \sum\limits_{j=1}^{n} a_{ij} \int_{-\infty}^{t} K_{ij}(t-s) g_j(v_j(s)) ds + I_i \\ \dot{v}_j(t) = -d_j v_j(t) + \sum\limits_{i=1}^{m} b_{ji} \int_{-\infty}^{t} H_{ji}(t-s) f_i(u_i(s)) ds + J_j \end{cases} \tag{1}$$

where $i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m;\ t > 0$; $u_i(t)$, $v_j(t)$ denote the potential of cell i and j at time t, respectively; $c_i,\ d_j$ are positive constants, they represent the rate with which the cell i and j reset their potential to the resting state when isolated from the other cells and inputs; $a_{ij},\ b_{ji}$ are real numbers, they denote the strengths of connectivity between the cells j and i; $f_i,\ g_j$ represent

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 727–734, 2010.

the signal propagation functions; I_i, J_j are the external inputs to cell i and j, respectively; $K_{ij}, H_{ji}, i = 1, \cdots, m, j = 1, \cdots, n$ are the delay kernels. Some BAM neural networks with distributed delays had been studied, see e.g. [3,5].

In the dynamical analysis for neural networks, most research focuses on the deterministic model. However, it should be pointed out that the circumstance noise may disturb the inputs of neural networks during their operating. The results in [6] suggested that the neural networks can be stabilized or destabilized by certain stochastic inputs. Some results about the stochastic stability of neural networks can be found in [7,8,9,10]. In these researches, the almost sure exponential stability problem of neural networks was studied by Blythe et al [7],Zhao and Ding [9],etc.

Motivated by above discussions, in this paper, we will study a stochastic BAM neural network with unbounded distributed delays described by some stochastic nonlinear integro-differential equations. With the help of suitable Lyapunov functional and the semimartingale convergence theorem, some sufficient conditions ensuring the almost sure exponential stability for the model are obtained. One numerical example is also given to illustrate our theoretic results in the end.

2 Model Formulation and Preliminaries

In this paper, we study the following stochastic BAM neural network with unbounded time delays:

$$\begin{cases} \mathrm{d}u_i = \Big[-c_i u_i + \sum\limits_{j=1}^{n} a_{ij} \int\limits_{-\infty}^{t} K_{ij}(t-s) g_j(v_j(s)) ds + I_i \Big] \mathrm{d}t + \sum\limits_{j=1}^{n} \rho_{ij}(v_j) \mathrm{d}w_{m+j} \\ \mathrm{d}v_j = \Big[-d_j v_j + \sum\limits_{i=1}^{m} b_{ji} \int\limits_{-\infty}^{t} H_{ji}(t-s) f_i(u_i(s)) ds + J_j \Big] \mathrm{d}t + \sum\limits_{i=1}^{m} \sigma_{ji}(u_i) \mathrm{d}w_i \end{cases} \tag{2}$$

where $w(t) = (w_1(t), w_2(t), ..., w_{m+n}(t))^T$ is an $(m+n)$-dimensional Brownian motion and $\sigma_{ij}(u_j)$, $\rho_{ji}(v_i)$ are noise intensity functions defined on R; the kernels $K_{ij}, H_{ji}, i = 1, \cdots, m, j = 1, \cdots, n$ are nonnegative functions defined on $[0, \infty)$, satisfying $\int_0^\infty K_{ij}(s)\mathrm{d}s = 1$, $\int_0^\infty K_{ij}(s)e^{\mu s}\mathrm{d}s < \infty$, $\int_0^\infty H_{ji}(s)\mathrm{d}s = 1$, $\int_0^\infty H_{ji}(s)e^{\nu s}\mathrm{d}s < \infty$, where μ and ν are positive real numbers. A typical expression of $K_{ij}(\cdot)(H_{ji}(\cdot))$ is $K_{ij}(s) = \frac{s^r}{r!}\lambda_{ij}^{r+1} e^{-\lambda_{ij} s}$, $s \in [0, \infty)$, where $\lambda_{ij} \in [0, \infty), r \in \{0, 1, \cdots, n\}$. About the kernels, the readers can refer to [3].

For the convenience of the stability analysis, we need the following assumptions:

(H1): The signal functions f_i, g_j, σ_{ji}, ρ_{ij} $i = 1, \cdots, m, j = 1, \cdots, n$ are Lipschitz continuous, that is, there exist positive constants α_i, β_j, γ_{ji}, η_{ij} such that

$$|f_i(z_1) - f_i(z_2)| \le \alpha_i |z_1 - z_2|, \quad |g_j(z_1) - g_j(z_2)| \le \beta_j |z_1 - z_2|,$$
$$|\sigma_{ji}(z_1) - \sigma_{ji}(z_2)| \le \gamma_{ji} |z_1 - z_2|, \quad |\rho_{ij}(z_1) - \rho_{ij}(z_2)| \le \eta_{ij} |z_1 - z_2|, \forall z_1, z_2 \in R.$$

(H2): Assume that (2) has an equilibrium $(u_i^*, v_j^*), i, j = 1, \cdots, n$, which satisfies

$$\begin{cases} -c_i u_i^* + \sum\limits_{j=1}^{n} a_{ij} g_j(v_j^*) ds + I_i = 0 \quad i = 1, \cdots, m \\ -d_j v_j^* + \sum\limits_{i=1}^{m} b_{ji} f_i(u_i^*) ds + J_j = 0, \quad j = 1, \cdots, n \end{cases} \tag{3}$$

(H3): For all $i = 1, \cdots, m, j = 1, \cdots, n$, $\sigma_{ji}(u_i^*) = 0$, $\rho_{ij}(v_j^*) = 0$.
Let $x_i(t) = u_i(t) - u_i^*$, $y_j(t) = v_j(t) - v_j^*$, then (2) becomes

$$\begin{cases} \mathrm{d}x_i = \Big[-c_i x_i + \sum\limits_{j=1}^{n} a_{ij} \int_{-\infty}^{t} K_{ij}(t-s) \tilde{g}_j(y_j(s)) ds \Big] \mathrm{d}t + \sum\limits_{j=1}^{n} \tilde{\rho}_{ij}(y_j) \mathrm{d}w_{m+j} \\ \mathrm{d}y_j = \Big[-d_j y_j + \sum\limits_{i=1}^{m} b_{ji} \int_{-\infty}^{t} H_{ji}(t-s) \tilde{f}_i(x_i(s)) ds \Big] \mathrm{d}t + \sum\limits_{i=1}^{m} \tilde{\sigma}_{ji}(x_i) \mathrm{d}w_i, \end{cases} \tag{4}$$

here $\tilde{f}_i(x_i(t)) = f_i(x_i(t) + u_i^*) - f_i(u_i^*)$, $\tilde{g}_j(y_j(t)) = g_j(y_j(t) + v_j^*) - g_j(v_j^*)$, $\tilde{\sigma}_{ji}(x_i(t)) = \sigma_{ji}(x_i(t) + u_i^*) - \sigma_{ji}(u_i^*)$, $\tilde{\rho}_{ij}(y_j(t)) = \rho_{ij}(y_j(t) + v_j^*) - \rho_{ij}(v_j^*)$. From (H1), one can easily see that $|\tilde{f}_i(x_i)| \le \alpha_i |x_i|$, $|\tilde{g}_j(y_j)| \le \beta_j |y_j|$, $|\tilde{\sigma}_{ji}(x_i)| \le \gamma_{ji} |x_i|$, $|\tilde{\rho}_{ij}(y_j)| \le \eta_{ij} |y_j|$, $\forall x_i, y_j \in R^n$.

Now, we give the definition of almost sure stability of stochastic system (4).

Definition 1. *The trivial solution of (4) is said to be almost surely exponentially stable if for almost all sample paths of the solution $(x(t), y(t))$, we have*

$$\limsup_{t \to \infty} \frac{1}{t} \ln |(x^T(t), y^T(t)))^T\| < 0.$$

The following Semimartingale convergence theorem plays a key role in proving the almost sure stability of several stochastic systems [7,9].

Lemma 1. *(Semimartingale convergence theorem [11]) Let $A(t)$ and $U(t)$ be two continuous adapted increasing processes on $t \ge 0$ with $A(0) = U(0) = 0$ a.s. Let $M(t)$ be a real-valued continuous local martingale with $M(0) = 0$ a.s. Let ξ be a nonnegative $\mathcal{F}_0$-measurable random variable. Define*

$$X(t) = \xi + A(t) - U(t) + M(t), \quad \text{for } t \ge 0$$

If $X(t)$ is nonnegative, then

$$\{\lim_{t \to \infty} A(t) < \infty\} \subset \{\lim_{t \to \infty} X(t) < \infty\} \cap \{\lim_{t \to \infty} U(t) < \infty\}, \text{ a.s.},$$

where $B \subset D$ a.s. means $P(B \cap D^c) = 0$. In particular, If $\lim\limits_{t \to \infty} A(t) < \infty$ a.s., then for almost all $\omega \in \Omega$

$$\lim_{t \to \infty} X(t) < \infty \quad \text{and} \quad \lim_{t \to \infty} U(t) < \infty,$$

that is both $X(t)$ and $U(t)$ converge to finite random variables.

3 Main Results

The purpose of this section is to study the stability of (4). By constructing a suitable Lyapunov functional and using the semimartingale convergence theorem, the almost sure stability criterion is derived as follows:

Theorem 1. *Assume that (H1)(H2)(H3) hold. If there exist constants $p_i > 0$, $q_j > 0$, such that the following inequalities hold for all $i = 1, \cdots, m, j = 1, \cdots, n$*

$$2c_i p_i > p_i \sum_{j=1}^{n} |a_{ij}| + \sum_{j=1}^{n} q_j \gamma_{ji}^2 + \alpha_i^2 \sum_{j=1}^{n} q_j |b_{ji}| \tag{5}$$

$$2d_j q_j > q_j \sum_{i=1}^{m} |b_{ji}| + \sum_{i=1}^{m} p_i \eta_{ij}^2 + \beta_j^2 \sum_{i=1}^{m} p_i |a_{ij}|, \tag{6}$$

then the trivial solution of (4) is almost surely exponentially stable.

Proof. It can be seen from (5)-(6) that there exists a positive constant λ, which satisfies $\lambda < \min\{\mu, \nu\}$, such that

$$\varphi_i(\lambda) = (2c_i - \lambda - \sum_{j=1}^{n} |a_{ij}|)p_i - \sum_{j=1}^{n} q_j \gamma_{ji}^2 - \alpha_i^2 \sum_{j=1}^{n} q_j |b_{ji}| \int_0^{\infty} H_{ji}(s) e^{\lambda s} \mathrm{d}s > 0 \tag{7}$$

$$\phi_j(\lambda) = (2d_j - \lambda - \sum_{i=1}^{m} |b_{ji}|)q_j - \sum_{i=1}^{m} p_i \eta_{ij}^2 - \beta_j^2 \sum_{i=1}^{m} p_i |a_{ij}| \int_0^{\infty} K_{ij}(s) e^{\lambda s} \mathrm{d}s > 0 \tag{8}$$

Constructing a Lyapunov functional candidate as follows:

$$\begin{aligned} V(t, x(t), y(t)) = e^{\lambda t} [\sum_{i=1}^{m} p_i x_i^2(t) + \sum_{j=1}^{n} q_j y_j^2(t)] \\ + \sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}| p_i \int_0^{\infty} \Big[K_{ij}(s) \int_{t-s}^{t} e^{\lambda(\theta+s)} \tilde{g}_j^2(y_j(\theta) \mathrm{d}\theta \Big] \mathrm{d}s \\ + \sum_{j=1}^{n} \sum_{i=1}^{m} |b_{ji}| q_j \int_0^{\infty} \Big[H_{ji}(s) \int_{t-s}^{t} e^{\lambda(\theta+s)} \tilde{f}_i^2(x_i(\theta) \mathrm{d}\theta \Big] \mathrm{d}s \end{aligned} \tag{9}$$

By employing Itô formula, we have

$$\begin{aligned} \mathrm{d}V(t, x(t), y(t)) = \mathcal{L}V(t, x(t), y(t))\mathrm{d}t + V_x(t, x(t), y(t))\Sigma_1(x(t))\mathrm{d}W^{(1)}(t) \\ + V_y(t, x(t), y(t))\Sigma_2(y(t))\mathrm{d}W^{(2)}(t), \end{aligned} \tag{10}$$

where

$$\mathcal{L}V(t, x(t), y(t))$$

$$
\begin{aligned}
&= \lambda e^{\lambda t}[\sum_{i=1}^{m} p_i x_i^2(t) + \sum_{j=1}^{n} q_j y_j^2(t)] \\
&\quad +2e^{\lambda t}\sum_{i=1}^{m} p_i x_i(t)\Big[-c_i x_i(t) + \sum_{j=1}^{n} a_{ij}\int_{-\infty}^{t} K_{ij}(t-s)\tilde{g}_j(y_j(s))ds\Big] \\
&\quad +2e^{\lambda t}\sum_{j=1}^{n} q_j y_j(t)\Big[-d_j y_j(t) + \sum_{i=1}^{n} b_{ji}\int_{-\infty}^{t} H_{ji}(t-s)\tilde{f}_i(x_i(s))ds\Big] \\
&\quad +e^{\lambda t}[\sum_{i=1}^{m}\sum_{j=1}^{n} p_i\tilde{\rho}_{ij}^2(y_j(t)) + \sum_{j=1}^{n}\sum_{i=1}^{m} q_j\tilde{\sigma}_{ji}^2(x_i(t))] \\
&\quad +e^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|p_i\int_0^{\infty} K_{ij}(s)e^{\lambda s}\mathrm{d}s\cdot\tilde{g}_j^2(y_j(t)) \\
&\quad -e^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|p_i\int_0^{\infty} K_{ij}(s)\tilde{g}_j^2(y_j(t-s))\mathrm{d}s \\
&\quad +e^{\lambda t}\sum_{j=1}^{n}\sum_{i=1}^{m}|b_{ji}|q_j\int_0^{\infty} H_{ji}(s)e^{\lambda s}\mathrm{d}s\cdot\tilde{f}_i^2(x_i(t)) \\
&\quad -e^{\lambda t}\sum_{j=1}^{n}\sum_{i=1}^{m}|b_{ji}|q_j\int_0^{\infty} H_{ji}(s)\tilde{f}_i^2(x_i(t-s))\mathrm{d}s \qquad (11)
\end{aligned}
$$

Noting that

$$
\begin{aligned}
&2\sum_{i=1}^{m}\sum_{j=1}^{n} p_i a_{ij} x_i(t)\int_{-\infty}^{t} K_{ij}(t-s)\tilde{g}_j(y_j(s))ds \\
&\le \sum_{i=1}^{m}\sum_{j=1}^{n} p_i|a_{ij}|x_i^2(t) + \sum_{i=1}^{m}\sum_{j=1}^{n} p_i|a_{ij}|\int_0^{\infty} K_{ij}(s)\tilde{g}_j^2(y_j(t-s))\mathrm{d}s, \qquad (12)
\end{aligned}
$$

$$
\begin{aligned}
&2\sum_{j=1}^{n}\sum_{i=1}^{m} q_j b_{ji} y_j(t)\int_{-\infty}^{t} H_{ji}(t-s)\tilde{f}_i(x_i(s))ds \\
&\le \sum_{j=1}^{n}\sum_{i=1}^{m} q_j|b_{ji}|y_j^2(t) + \sum_{j=1}^{n}\sum_{i=1}^{m} q_j|b_{ji}|\int_0^{\infty} H_{ji}(s)\tilde{f}_i^2(x_i(t-s))\mathrm{d}s, \qquad (13)
\end{aligned}
$$

combing inequalities $\tilde{f}_i^2(x_i(t)) \le \alpha_i^2 x_i^2(t)$, $\tilde{g}_j^2(y_j(t)) \le \beta_j^2 y_j^2(t)$, $\tilde{\sigma}_{ji}^2(x_i(t)) \le \gamma_{ji}^2 x_i^2(t)$, $\tilde{\rho}_{ij}^2(y_j(t)) \le \eta_{ij}^2 y_j^2(t)$, it can be derived that

$$
\begin{aligned}
\mathcal{L}V(t,x(t),y(t)) &\le e^{\lambda t}[\sum_{i=1}^{m}(\lambda p_i - 2c_i p_i)x_i^2(t) + \sum_{j=1}^{n}(\lambda q_j - 2d_j q_j)y_j^2(t)] \\
&\quad +e^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}\Big[|a_{ij}|p_i x_i^2(t) + |b_{ji}|q_j y_j^2(t)\Big]
\end{aligned}
$$

$$
\begin{aligned}
&+e^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}\left[p_i\eta_{ij}^2y_j^2(t)+q_j\gamma_{ji}^2x_i^2(t)\right]\\
&+e^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|p_i\int_0^{\infty}K_{ij}(s)e^{\lambda s}\mathrm{d}s\cdot\beta_j^2y_j^2(t)\\
&+e^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}|b_{ji}|q_j\int_0^{\infty}H_{ji}(s)e^{\lambda s}\mathrm{d}s\cdot\alpha_i^2x_i^2(t)\\
&=-e^{\lambda t}[\sum_{i=1}^{m}\varphi_i(\lambda)x_i^2(t)+\sum_{j=1}^{n}\phi_i(\lambda)y_i^2(t)]. \qquad (14)
\end{aligned}
$$

Using Itô formula again, for all $T>0$

$$
\begin{aligned}
&V(T,x(T),y(T))\\
&=V(0)+\int_0^T\mathcal{L}V(t,x(t),y(t))\mathrm{d}t+\int_0^TV_x(t,x,y)\Sigma_1(x(t))\mathrm{d}W^{(1)}(t)\\
&\quad+\int_0^TV_y(t,x,y)\Sigma_2(y(t))\mathrm{d}W^{(2)}(t)\\
&\le V(0)-\int_0^Te^{\lambda t}[\sum_{i=1}^{m}\varphi_i(\lambda)x_i^2(t)+\sum_{j=1}^{n}\phi_i(\lambda)y_i^2(t)]dt+2\int_0^Te^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}p_i\\
&\quad\times x_i(t)\tilde{\rho}_{ij}(y_j(t))dw_{n+j}(t)+2\int_0^Te^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}q_jy_j(t)\tilde{\sigma}_{ji}(x_i(t))dw_i(t), \qquad (15)
\end{aligned}
$$

where

$$
\begin{aligned}
V(0)&\le\sum_{i=1}^{m}p_ix_i^2(0)+\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|p_i\int_0^{\infty}K_{ij}(s)e^{\lambda s}\mathrm{d}s\cdot\frac{1}{\lambda}\beta_j^2\sup_{-\infty<s\le0}\{y_j^2(s)\}\\
&\quad+\sum_{j=1}^{n}q_jy_j^2(0)+\sum_{j=1}^{n}\sum_{i=1}^{m}|b_{ji}|q_j\int_0^{\infty}H_{ji}(s)e^{\lambda s}\mathrm{d}s\cdot\frac{1}{\lambda}\alpha_i^2\sup_{-\infty<s\le0}\{x_i^2(s)\}\\
&<+\infty. \qquad (16)
\end{aligned}
$$

It can be seen that the right hand-side of the above inequality is a nonnegative semimartingale, and from Lemma 1 one gets

$$
\lim_{T\to\infty}X(T)<\infty\quad\text{a.s.} \qquad (17)
$$

here

$$
\begin{aligned}
X(T)&=V(0,x(0),y(0))+2\int_0^Te^{\lambda t}\sum_{i=1}^{m}\sum_{j=1}^{n}p_i\tilde{\rho}_{ij}(y_j(t))dw_{m+j}(t)\\
&\quad+2\int_0^Te^{\lambda t}\sum_{j=1}^{n}\sum_{i=1}^{m}q_j\tilde{\sigma}_{ji}(x_i(t))dw_i(t). \qquad (18)
\end{aligned}
$$

Moreover

$$\lim_{T\to\infty} e^{\lambda T}[\sum_{i=1}^{m} p_i x_i^2(T) + \sum_{j=1}^{n} q_j y_j^2(T)] \le \lim_{T\to\infty} V(T, x(T), y(T)) < \infty \quad \text{a.s.} \tag{19}$$

So

$$\limsup_{T\to\infty} \frac{1}{T} \log \|(x^T(T), y^T(T))^T\| \le -\lambda, \tag{20}$$

and then the trivial solution of (4) is almost surely exponentially stable.

Corollary 1. *Assume that (H1)(H2) hold. If there exist constants $p_i > 0$, $q_j > 0$, such that the following inequalities hold for all $i = 1, \cdots, m, j = 1, \cdots, n$*

$$2c_i p_i > p_i \sum_{j=1}^{n} |a_{ij}| + \alpha_i^2 \sum_{j=1}^{n} q_j |b_{ji}| \tag{21}$$

$$2d_j q_j > q_j \sum_{i=1}^{m} |b_{ji}| + \beta_j^2 \sum_{i=1}^{m} p_i |a_{ij}|, \tag{22}$$

then the trivial solution of (4) is exponentially stable.

4 Numerical Example

Consider the following BAM neural network:

$$\begin{cases} \mathrm{d}x_i = \Big[-c_i x_i + \sum\limits_{j=1}^{2} a_{ij} \int_{-\infty}^{t} K_{ij}(t-s) g_j(y_j(s)) ds \Big] \mathrm{d}t + \sum\limits_{j=1}^{2} \rho_{ij}(y_j) \mathrm{d}w_{2+j} \\ \mathrm{d}y_j = \Big[-d_j y_j + \sum\limits_{i=1}^{2} b_{ji} \int_{-\infty}^{t} H_{ji}(t-s) f_i(x_i(s)) ds \Big] \mathrm{d}t + \sum\limits_{i=1}^{2} \sigma_{ji}(x_i) \mathrm{d}w_i, \end{cases} \tag{23}$$

here $i, j = 1, 2$. Taking the kernels as

$$K(s) = \begin{pmatrix} e^{-s} & e^{-s} \\ e^{-s} & e^{-s} \end{pmatrix}, \qquad H(s) = \begin{pmatrix} 2e^{-2s} & 2e^{-2s} \\ 2e^{-2s} & 2e^{-2s} \end{pmatrix}, \tag{24}$$

the signal functions as $f_1(x_1) = f_2(x_2) = \tanh(x_1)$, $g_1(y_1) = g_2(y_2) = \frac{1}{2}(|y_1 + 1| - |y_1 - 1|)$, the noise intensity functions as

$$\Sigma^{(1)} = \begin{pmatrix} \sigma_{11}(x_1) & \sigma_{12}(x_2) \\ \sigma_{21}(x_1) & \sigma_{22}(x_2) \end{pmatrix} = \begin{pmatrix} 0.2x_1 & 0.1x_2 \\ 0.1x_1 & 0.1x_2 \end{pmatrix}, \tag{25}$$

$$\Sigma^{(2)} = \begin{pmatrix} \rho_{11}(y_1) & \rho_{12}(y_2) \\ \rho_{21}(y_1) & \rho_{22}(y_2) \end{pmatrix} = \begin{pmatrix} 0.2y_1 & 0.1y_2 \\ 0 & 0.2y_2 \end{pmatrix}. \tag{26}$$

The system parameters are selected as:

$$C = \begin{pmatrix} 2.0 & 0 \\ 0 & 2.4 \end{pmatrix}, A = \begin{pmatrix} 1.4 & -0.5 \\ 0.3 & 1.7 \end{pmatrix}, D = \begin{pmatrix} 2.2 & 0 \\ 0 & 2.6 \end{pmatrix}, B = \begin{pmatrix} 0.9 & 0.2 \\ -0.2 & 1.1 \end{pmatrix}. \tag{27}$$

If we further take $p_1 = 1.2$, $p_2 = 0.9$, $q_1 = 1.7$, $q_2 = 1.4$, it is easy to verify that (5) and (6) are satisfied. From Theorem 1, (4) is almost surely exponentially stable. Letting $\lambda = 0.3$, (7), (8) hold, so we can obtain (20), that is, the convergent rate of (23) is not smaller than 0.3.

5 Conclusions

In this paper, we have studied the almost sure stability problem for stochastic BAM neural networks. Considering the complex mechanism of signal propagation in a large scale neural network, we have modeled the BAM neural networks with distributed time delays. During the proof of the stability of our system, the well-known semimartingale convergence theorem plays an important role. In the end, a suitable example has been provided to illustrate the theoretical results, and the trajectories of our system are also plotted which verify the stability of the system directly.

References

1. Kosko, B.: Adaptive bidirectional associative memories. Appl. Optics. 26, 4947–4960 (1987)
2. Kosko, B.: Bidirectional associative memories. IEEE Trans. Syst. Man Cybern. 18(1), 49–60 (1988)
3. Gopalsamy, K., He, X.: Delay-independent stability in bidirectional associative memory networks. IEEE Trans. Neural Netw. 5, 998–1002 (1994)
4. Liang, J., Cao, J.: Exponential stability of continuous-time and discrete-time bidirectional associative memory networks with delays. Chaos Solitons Fractals 22(4), 773–785 (2004)
5. Song, Q., Cao, J.: Global exponential stability of bidirectional associative memory neural networks with distributed delays. J. Computat. Appl. Math. 202, 266–279 (2007)
6. Liao, X., Mao, X.: Exponential stability and instability of stochastic neural network. Stochastic Analysis and Applications 14, 165–185 (1996)
7. Blythe, S., Mao, X., Liao, X.: Stability of stochastic delay neural networks. Journal of the Franklin Institute 338, 481–495 (2001)
8. Wang, Z., Liu, Y., Li, M., Liu, X.: Stability analysis for stochastic Cohen-Grossberg neural networks with mixed time delays. IEEE Trans. Neural Netw. 17, 814–820 (2006)
9. Zhao, H., Ding, N.: Dynamic analysis of stochastic bidirectional associative memory neural networks with delays. Chaos Solitons Fractals 32, 1692–1702 (2007)
10. Wang, G., Cao, J.: Stability analysis for stochastic BAM neural networks with Markovian jumping parameters. Neurocomputing 72, 3901–3906 (2009)
11. Mao, X.: Stochastic Differential Equation and Application. Horwood Publishing, Chichester (1997)

Dissipativity in Mean Square of Non-autonomous Impulsive Stochastic Neural Networks with Delays

Zhiguo Yang[1] and Zhichun Yang[2]

[1] College of Mathematics and Software Science, Sichuan Normal University, Chengdu 610066, China
[2] Department of Mathematics, Chongqing Normal University, Chongqing 400047, China

Abstract. In this paper, we study non-autonomous impulsive stochastic neural networks with delays. By establishing an impulsive differential inequality and an L-operator differential inequality and using stochastic analysis technique, we obtain some sufficient conditions ensuring the dissipativity in mean square of non-autonomous impulsive stochastic neural networks with delays. The results extend and improve those of the earlier publications. An example is also discussed to illustrate the efficiency of the obtained results.

Keywords: Impulsive, Stochastic, Neural networks, Dissipativity.

1 Introduction

Neural networks play a very important role in associative memories, pattern recognition and optimization, etc. Therefore, the theory of neural networks has been developed very quickly. In recent years, stochastic delayed neural networks and impulsive delayed neural networks have attracted considerable attention. Many interesting results on the asymptotic behaviors of stochastic delayed neural networks [1] and impulsive delayed neural networks [2] have been reported, respectively. It is well known that impulsive effects and stochastic effects frequently coexist in neural networks. Therefore, it is necessary to investigate the asymptotic behaviors of impulsive stochastic neural networks with delays. However, to the best of our knowledge, there are few results on this problem [3]-[4], since the corresponding theory for impulsive stochastic neural networks has not yet been developed.

As is well known that stability analysis of neural networks is a prerequisite for the practice design and applications. Therefore, the stability of neural networks has received much attention. Nevertheless, from a practical point of view, it is not always the case that there exists an equilibrium point for every neural network, especially for non-autonomous impulsive stochastic neural networks. Therefore, it is necessary to investigate the dissipativity of neural networks which has found applications in diverse areas such as stability theory, chaos and synchronization

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 735–744, 2010.

theory, system norm estimation, and robust control. On the base of the above discussion, this paper is devoted to the discussion of the dissipativity of non-autonomous impulsive stochastic neural networks with delays.

2 Model and Preliminaries

Let I denote the n-dimensional unit matrix, $\mathcal{N} \triangleq \{1, 2, \cdots, n\}$, $\mathbb{N} \triangleq \{1, 2, \cdots\}$. $C(X, Y)$ denote the space of continuous mappings from the space X to Y.

$PC(J, R^n) \triangleq \{\psi : J \to R^n \mid \psi(t)$ is continuous for all but at most countable points $s \in J$ and at these points $s \in J$, $\psi(s^+)$ and $\psi(s^-)$ exist, $\psi(s^+) = \psi(s)\}$, where $J \subset R$ is an interval, $\psi(s^+)$ and $\psi(s^-)$ denote the right-hand and left-hand limits of the function $\psi(s)$, respectively.

For $\varphi(t) \in C(J, R^n)$ or $\varphi(t) \in PC(J, R^n)$, we define

$$[\varphi(t)]_\tau = ([\varphi_1(t)]_\tau, \cdots, [\varphi_n(t)]_\tau)^T,\ [\varphi_i(t)]_\tau = \sup_{-\tau \le s \le 0} \{\varphi_i(t+s)\},\ i \in \mathcal{N}, \quad (1)$$

where τ is a positive constant. Let $D^+\varphi(t)$ denote the upper right derivative of $\varphi(t)$ at t.

Let $(\Omega, \mathscr{F}, \{\mathscr{F}_t\}_{t\ge 0}, P)$ be a complete probability space with a filtration $\{\mathscr{F}_t\}_{t\ge 0}$ satisfying the usual conditions (i.e., it is right continuous and $\mathscr{F}_0$ contains all P-null sets). $w(t) = (w_1(t), \cdots, w_m(t))^T$ is an m-dimensional Brownian motion defined on $(\Omega, \mathscr{F}, \{\mathscr{F}_t\}_{t\ge 0}, P)$. Let $PC^b_{\mathscr{F}_0}([-\tau, 0], R^n)$ denote the family of all bounded $\mathscr{F}_0$-measurable, $PC([-\tau, 0], R^n)$-valued random variables φ, satisfying $\|\varphi\|^2_{L^2} = \sup_{-\tau\le\theta\le 0} \mathbb{E}|\varphi(\theta)|^2 < \infty$, where $\mathbb{E}$ denote the expectation of a stochastic process, $|.|$ is Euclidean norm of R^n. Let $\sigma = \sigma(t, x, y) = (\sigma_{il}(t, x_i, y_i))_{n\times m},\ \sigma : R\times R^n \times R^n \to R^{n\times m}$, $\sigma_i = \sigma_i(t, x_i, y_i) = (\sigma_{i1}(t, x_i, y_i), \cdots, \sigma_{im}(t, x_i, y_i))$ be ith row vector of $\sigma(t, x, y),\ i \in \mathcal{N}$.

In this paper, we consider impulsive stochastic delayed neural networks

$$\begin{cases} dx_i(t) = [-a_i(t)x_i(t) + \sum_{j=1}^{n} b_{ij}(t) f_j(x_j(t-\tau_{ij}(t))) + J_i(t)]dt \\ \qquad\qquad + \sum_{l=1}^{m} \sigma_{il}(t, x_i(t), x_i(t-r_i(t)))dw_l(t),\ t \ge t_0,\ t \ne t_k,\ i \in \mathcal{N}, \\ x_i(t) = I_{ik}(x_1(t^-), \cdots, x_n(t^-)),\ t \ge t_0,\ t = t_k,\ i \in \mathcal{N}, \\ x_i(t_0 + s) = \varphi_i(s),\ -\tau \le s \le 0,\ i \in \mathcal{N}, \end{cases} \quad (2)$$

where $a_i(t)$, $b_{ij}(t)$, $J_i(t)$, $\tau_{ij}(t)$, $r_i(t)$, $f_j(t) \in C(R, R)$, $\tau_{ij}(t), r_i(t) \in [-\tau, 0]$, $i, j \in \mathcal{N}$. And the initial function $\varphi(s) = (\varphi_1(s), \cdots, \varphi_n(s))^T \in PC^b_{\mathscr{F}_0}([-\tau, 0], R^n)$, the impulsive function $I_k = (I_{1k}, \cdots, I_{nk})^T \in C(R^n, R^n)$, $k \in \mathbb{N}$, and the fixed impulsive moments t_k satisfy $t_1 < t_2 < \cdots$, $\lim_{k\to\infty} t_k = \infty$.

Throughout this paper, we assume that for any $\varphi \in PC^b_{\mathscr{F}_0}([-\tau, 0], R^n)$, there exists at least one solution of (1), which is denoted by $x(t, t_0, \varphi)$ or $x(t)$.

Definition 1. The networks (2) are said to be dissipative in mean square, if there is a constant $B > 0$, such that for any solution $x(t) = (x_1(t), \cdots, x_n(t))^T$ of networks (1), one has

$$\lim_{t\to\infty} \sup \mathbb{E}x_i^2(t) \le B, \quad i \in \mathcal{N}. \tag{3}$$

Definition 2. The networks (2) are said to be exponentially dissipative in mean square, if there are positive constants M_1, M_2 and λ such that for any solution $x(t, t_0, \varphi)$, one has

$$\mathbb{E}|x(t, t_0, \varphi)|^2 \le M_1 \|\varphi\|_{L^2}^2 e^{-\lambda(t-t_0)} + M_2, \quad t \ge t_0. \tag{4}$$

The zero solution of (1) is said to be exponentially stable in mean square when $M_2 = 0$.

For an M-matrix S [5], we define $\Omega_M(S) \triangleq \{z \in R^n \mid Sz > 0,\ z > 0\}$.

Lemma 1. [5] *For an M-matrix S, $\Omega_M(S)$ is nonempty and satisfies,*

$$k_1 z_1 + k_2 z_2 \in \Omega_M(S), \ \text{for any } k_1,\ k_2 > 0,\ z_1,\ z_2 \in \Omega_M(S).$$

3 Differential Inequality

In this section, we will first establish an impulsive differential inequality with delays and then introduce an L-operator differential inequality with delays.

Theorem 1. *For $b \in (t_0, +\infty]$, let $u(t) = (u_1(t), u_2(t), \cdots, u_n(t))^T \in C([t_0, b), R^n)$ be a solution of the following inequality with the initial condition $u(\theta) \in PC([t_0 - \tau, t_0], R^n)$*

$$D^+u(t) \le h(t)[Pu(t) + Q[u(t)]_\tau + J], \quad t \in [t_0, b), \tag{5}$$

where $P = (p_{ij})_{n\times n}$, $p_{ij} \ge 0$ for $i \ne j$, $Q = (q_{ij})_{n\times n} \ge 0$, $J = (J_1, \cdots, J_n)^T \ge 0$, $h(t)$ is a nonnegative integral function and $\sup\limits_{t\ge t_0} \int_{t-\tau}^{t} h(s)ds \le H < \infty$. If $S = -(P + Q)$ is an M-matrix, then

$$u(t) \le ze^{-\lambda \int_{t_0}^{t} h(s)ds} - (P + Q)^{-1}J, \quad t \in [t_0, b) \tag{6}$$

provided that the initial conditions satisfies

$$u(\theta) \le ze^{-\lambda \int_{t_0}^{\theta} h(s)ds} - (P + Q)^{-1}J, \quad t_0 - \tau \le \theta \le t_0, \tag{7}$$

where $z = (z_1, \cdots, z_n)^T \in \Omega_M(S)$ and the positive constant λ satisfies

$$[P + Qe^{\lambda H} + \lambda I]z < 0. \tag{8}$$

Proof. Since $S = -(P+Q)$ is an M-matrix, there exists a vector $z = (z_1, \cdots, z_n)^T \in \Omega_M(S)$ such that $(P+Q)z < 0$. By using continuity, we know that there must exist a $\lambda > 0$ such that (8) holds, that is,

$$\sum_{j=1}^{n}(p_{ij} + q_{ij}e^{\lambda H})z_j < -\lambda z_i, \quad i \in \mathcal{N}. \tag{9}$$

Let $L = -(P+Q)^{-1}J$, $L = (L_1, \cdots, L_n)^T$. We have $(P+Q)L + J = 0$, or

$$\sum_{j=1}^{n}(p_{ij} + q_{ij})L_j + J_i = 0, \quad i \in \mathcal{N}. \tag{10}$$

For the above λ and τ, there is a positive constant ε_0 such that

$$e^{\lambda\tau\varepsilon} < 1 + \varepsilon, \quad \text{for all } \varepsilon \in (0,\ \varepsilon_0). \tag{11}$$

We at first shall prove that for any positive constant $\varepsilon \in (0,\ \varepsilon_0)$,

$$u_j(t) \le (1+\varepsilon)z_j e^{-\lambda\int_{t_0}^{t}[h(s)-\varepsilon]ds} + L_j \triangleq y_j(t), \quad t \in [t_0, b),\ j \in \mathcal{N}. \tag{12}$$

By (7) and (11), we have

$$u_j(\theta) < (1+\varepsilon)z_j e^{-\lambda\int_{t_0}^{\theta}[h(s)-\varepsilon]ds} + L_j = y_j(\theta),\ \theta \in [t_0-\tau, t_0],\ j \in \mathcal{N}. \tag{13}$$

If the inequality (12) is not true, then there must be constant $t_1 \in (t_0, b)$ and some integer $m \in \mathcal{N}$ such that

$$u_m(t_1) = y_m(t_1), \qquad D^+u_m(t_1) \ge D^+y_m(t_1), \tag{14}$$

$$u_j(t) \le y_j(t), \quad t \in [t_0-\tau, t_1], \quad j \in \mathcal{N}. \tag{15}$$

From (15), we have

$$\begin{aligned}[u_j(t)]_\tau &\le (1+\varepsilon)z_j e^{-\lambda\int_{t_0}^{t-\tau}h(s)ds}e^{\lambda\varepsilon(t-t_0)} + L_j \\ &\le (1+\varepsilon)z_j e^{-\lambda\int_{t_0}^{t}[h(s)-\varepsilon]ds}e^{\lambda H} + L_j, \quad t \in [t_0, t_1], \quad j \in \mathcal{N}.\end{aligned} \tag{16}$$

By using (5), (9), (10), (14), (15), (16) and $p_{ij} \ge 0\,(i \ne j)$, $q_{ij} \ge 0$, we have

$$\begin{aligned}D^+u_m(t_1) &\le h(t_1)\{\sum_{j=1}^{n}p_{mj}[(1+\varepsilon)z_j e^{-\lambda\int_{t_0}^{t_1}[h(s)-\varepsilon]ds} + L_j] \\ &\quad + \sum_{j=1}^{n}q_{mj}[(1+\varepsilon)z_j e^{-\lambda\int_{t_0}^{t_1}[h(s)-\varepsilon]ds}e^{\lambda H} + L_j] + J_m\} \\ &= h(t_1)\sum_{j=1}^{n}(p_{mj} + q_{mj}e^{\lambda H})z_j(1+\varepsilon)e^{-\lambda\int_{t_0}^{t_1}[h(s)-\varepsilon]ds} \\ &\le -\lambda h(t_1)z_m(1+\varepsilon)e^{-\lambda\int_{t_0}^{t_1}[h(s)-\varepsilon]ds} \\ &< -\lambda[h(t_1)-\varepsilon]z_m(1+\varepsilon)e^{-\lambda\int_{t_0}^{t_1}[h(s)-\varepsilon]ds} \\ &= D^+y_m(t_1)\end{aligned} \tag{17}$$

which contradicts the inequality in (14). Thus (12) holds for all $t \in [t_0, b)$, $\varepsilon \in (0, \varepsilon_0)$. Therefore, letting $\varepsilon \to 0$, we have

$$X(t) \le ze^{-\lambda \int_{t_0}^{t} h(s)ds} - (P+Q)^{-1}J, \quad t \in [t_0, b), \tag{18}$$

and the proof is completed.

Remark 1. The above Theorem 1 with $h(t) \ge 0$ is a natural generalization of some known results, for example, Theorem 3.1 in [2] when $h(t) \equiv 1$ and Lemma 5 in [6] when $h(t) > 0$, $b = \infty$, $PC([t_0 - \tau, t_0], R^n)$ replaced by $C([t_0 - \tau, t_0], R^n)$.

For the well-known $\mathcal{L}$-operator given by the $Itô's$ formula, by the above Theorem 1, we have the following Theorem.

Theorem 2. *Let $J = (J_1, \cdots, J_n)^T \ge 0$, $P = (p_{ij})_{n\times n}$ and $p_{ij} \ge 0$ for $i \ne j$, $Q = (q_{ij})_{n\times n} \ge 0$ and $S = -(P+Q)$ be an M-matrix, $h(t)$ be a nonnegative integral function satisfying $\sup\limits_{t\ge t_0} \int_{t-\tau}^{t} h(s)ds \le H < \infty$. Assume that there exist functions $V_i(x) \in C^2(R^n, R)$ such that for $t \in [t_{k-1}, t_k)$,*

$$\mathbb{E}\mathcal{L}V_i(x(t)) \le h(t)[\sum_{j=1}^{n} p_{ij}\mathbb{E}V_j(x(t)) + \sum_{j=1}^{n} q_{ij}[\mathbb{E}V_j(x(t))]_\tau + J_i],\ i \in \mathcal{N}. \tag{19}$$

Then we have

$$\mathbb{E}V_i(x(t)) \le z_i e^{-\lambda \int_{t_0}^{t} h(s)ds} + L_i,\ t \in [t_{k-1}, t_k),\ i \in \mathcal{N}, \tag{20}$$

provided that the initial conditions satisfy

$$\mathbb{E}V_i(x(t)) \le z_i e^{-\lambda \int_{t_0}^{t} h(s)ds} + L_i,\ \ t \in [t_{k-1} - \tau, t_{k-1}],\ i \in \mathcal{N}, \tag{21}$$

where $L = (L_1, \cdots, L_n)^T = -(P+Q)^{-1}J$, $z = (z_1, \cdots, z_n)^T \in \Omega_M(S)$, and the positive number λ satisfies the following inequality

$$[\lambda I + P + Qe^{\lambda H}]z < 0\ . \tag{22}$$

Proof. The proof is similar with one of Theorem 3.1 in [5]. We omit it.

Remark 2. The above Theorem 2 is Theorem 3.1 in [5] when $h(t) \equiv 1, J = 0$.

4 Dissipativity in Mean Square

Here, we firstly introduce the following assumptions.
(A_1) There exist constants $\hat{a}_i > 0$, $\hat{b}_{ij} \ge 0$, $\hat{J}_i \ge 0$, $v_j \ge 0$ and a nonnegative integral function $h(t)$ satisfying $\sup\limits_{t\ge t_0} \int_{t-\tau}^{t} h(s)ds \le H < \infty$ such that

$$a_i(t) \geq \hat{a}_i h(t),\ |b_{ij}(t)| \leq h(t)\hat{b}_{ij},\ |J_i(t)| \leq h(t)\tilde{J}_i,\ |f_j(x_j)| \leq v_j|x_j|,\ \ i,j \in \mathcal{N}.$$

(A_2) There exist nonnegative constants $c_i, d_i, \bar{J}_i$ such that the ith row vector σ_i of σ in (2) satisfies

$$\mathbb{E}[\sigma_i \sigma_i^T] \leq h(t)(c_i \mathbb{E}x_i^2(t) + d_i[\mathbb{E}x_i^2(t)]_\tau + \bar{J}_i),\ i \in \mathcal{N}. \tag{23}$$

(A_3) Let $S = -(P+Q)$ be an M-matrix, where

$$P = (p_{ij})_{n\times n},\ p_{ii} = -2\hat{a}_i + \sum_{j=1}^{n} \hat{b}_{ij}v_j + c_i + \tilde{J}_i,\ p_{ij} = 0,\ i \neq j, \tag{24}$$

$$Q = (q_{ij})_{n\times n},\ q_{ii} = d_i + \hat{b}_{ii}v_i,\ q_{ij} = \hat{b}_{ij}v_j,\ i \neq j,\ i,j \in \mathcal{N}. \tag{25}$$

(A_4) There exist nonnegative matrices $R_k = (R_{k_{ij}})_{n\times n}$ such that

$$[I_k(x)]^+ \leq R_k[x]^+,\ \ x \in R^n,\ \ k \in \mathbb{N}, \tag{26}$$

where $[I_k(x)]^+ = (|I_{1k}(x)|, \cdots, |I_{nk}(x)|)^T, [x]^+ = (|x_1|, \cdots, |x_n|)^T$. (A_5) There exist nonnegative constants δ and μ such that

$$\frac{\ln \delta_k}{t_k - t_{k-1}} \leq \delta,\ \ \mu = \sum_{k=1}^{\infty} \ln \mu_k < \infty\ \ k \in \mathbb{N}, \tag{27}$$

where constants $\delta_k \geq 1, \mu_k \geq 1$ satisfy

$$\delta_k z \geq \hat{R}_k z,\ \ \mu_k L \geq \hat{R}_k L,\ \ k \in \mathbb{N}, \tag{28}$$

where $z = (z_1, \cdots, z_n)^T \in \Omega_M(S),\ L = (L_1, \cdots, L_n)^T = -(P+Q)^{-1}\hat{J},\ \hat{J} = (\hat{J}_1, \cdots, \hat{J}_n)^T,\ \hat{J}_i = \tilde{J}_i + \bar{J}_i, i \in \mathcal{N}$,

$$\hat{R}_k = (\hat{R}_{k_{ij}})_{n\times n}, \quad \hat{R}_{k_{ij}} \geq R_{k_{ij}} \sum_{j=1}^{n} R_{k_{ij}}. \tag{29}$$

Theorem 3. *If (A_1)-(A_5) hold, then the solution $x_i(t)$ satisfies*

$$\mathbb{E}x_i^2(t) \leq\ z_i\, ||\varphi||_{L^2}^2 e^{-\int_{t_0}^{t}[\lambda h(s) - \delta]ds} + e^{\mu}L_i,\ t \geq t_0,\ i \in \mathcal{N}, \tag{30}$$

where the scalar $\lambda > 0$ satisfies the following inequality

$$[\lambda I + P + Qe^{\lambda H}]z < 0. \tag{31}$$

Proof. Since $S = -(P+Q)$ is an M-matrix, there exists a vector $z \in \Omega_M(S)$ satisfying $z \geq (1, \cdots, 1)^T \in R^n$ such that

$$Sz > 0 \quad \text{or} \quad (P+Q)z < 0. \tag{32}$$

By using continuity, there is a positive constant λ satisfying (31).

Let $V_i(x(t)) = x_i^2(t),\ i \in \mathcal{N}$, where $x(t) = (x_1(t), \cdots, x_n(t))^T$ is the solution of (1). Then, by the conditions (A_1)-(A_3), we can get

$$\begin{aligned}
&\mathbb{E}\mathcal{L}V_i(x)\\
&= 2\mathbb{E}[-a_i(t)x_i^2(t) + x_i(t)\sum_{j=1}^{n} b_{ij}(t)f_j(x_j(t-\tau_{ij}(t))) + J_i(t)x_i(t)]\\
&\quad +\mathbb{E}[\delta_i(t, x_i(t), x_i(t-r_i(t)))\delta_i^T(t, x_i(t), x_i(t-r_i(t)))]\\
&\le 2h(t)\mathbb{E}[-\hat{a}_i x_i^2(t) + \sum_{j=1}^{n} \hat{b}_{ij}v_j|x_i(t)x_j(t-\tau_{ij}(t))| + \tilde{J}_i|x_i(t)|]\\
&\quad +h(t)(c_i\mathbb{E}x_i^2(t) + d_i[\mathbb{E}x_i^2(t)]_\tau + \bar{J}_i)\\
&\le h(t)\mathbb{E}[-2\hat{a}_i\mathbb{E}x_i^2(t) + \sum_{j=1}^{n} \hat{b}_{ij}v_j\mathbb{E}x_i^2(t) + \sum_{j=1}^{n} \hat{b}_{ij}v_j[\mathbb{E}x_j^2(t)]_\tau + \tilde{J}_i\mathbb{E}x_i^2(t)\\
&\quad +\tilde{J}_i] + h(t)(c_i\mathbb{E}x_i^2(t) + d_i[\mathbb{E}x_i^2(t)]_\tau + \bar{J}_i)\\
&= h(t)(p_{ii}\mathbb{E}V_i(x(t)) + \sum_{j=1}^{n} q_{ij}[\mathbb{E}V_j(x(t))]_\tau + \hat{J}_i),\ t \in [t_{k-1}, t_k), i \in \mathcal{N}. \qquad (33)
\end{aligned}$$

For the initial condition $\varphi \in PC^b_{\mathscr{F}_0}([-\tau, 0], R^n)$, we can get

$$\mathbb{E}V_i(x(t)) \le z_i\|\varphi\|^2_{L^2}e^{-\lambda\int_{t_0}^{t} h(s)ds} + L_i,\ t \in [t_0 - \tau, t_0],\ i \in \mathcal{N}, \qquad (34)$$

where, $z = (z_1, \cdots, z_n)^T \in \Omega_M(S)$ is the one in (32), $L = (L_1, \cdots, L_n)^T \ge 0$ since S is an M-matrix.

From Lemma 1 and $z = (z_1, \cdots, z_n)^T \in \Omega_M(S)$, we have $\|\varphi\|^2_{L^2}z \in \Omega_M(S)$. Then, all conditions of Theorem 2 are satisfied by (33), (34) and (A_3). So

$$\mathbb{E}V_i(x(t)) \le z_i\|\varphi\|^2_{L^2}e^{-\lambda\int_{t_0}^{t} h(s)ds} + L_i,\ t \in [t_0, t_1),\ i \in \mathcal{N}, \qquad (35)$$

Suppose that for all $m = 1, \cdots, k$, the inequalities

$$\begin{aligned}
\mathbb{E}V_i(x(t)) \le\ & \delta_0\delta_1\cdots\delta_{m-1}z_i\|\varphi\|^2_{L^2}e^{-\lambda\int_{t_0}^{t} h(s)ds}\\
&+\mu_0\mu_1\cdots\mu_{m-1}L_i,\ t \in [t_{m-1}, t_m),\ i \in \mathcal{N}, \qquad (36)
\end{aligned}$$

hold, where $\delta_0 = 1, \mu_0 = 1$. Then from $(A4)$, (28), (29) and (36), we have

$$\mathbb{E}V_i(x(t_k)) = \mathbb{E}x_i^2(t_k) = \mathbb{E}|I_{ik}(x(t_k^-))|^2 \le \mathbb{E}\left(\sum_{j=1}^{n} R_{k_{ij}}|x_j(t_k^-)|\right)^2$$

$$\leq \mathbb{E}\left[(\sum_{j=1}^{n} R_{k_{ij}})\sum_{j=1}^{n} R_{k_{ij}}|x_j(t_k^-)|^2\right] \leq \sum_{j=1}^{n}\hat{R}_{k_{ij}}\mathbb{E}V_j(x(t_k^-))$$

$$\leq \delta_0\delta_1\cdots\delta_{k-1}\,||\varphi||^2_{L^2}\,e^{-\lambda\int_{t_0}^{t}h(s)ds}\sum_{j=1}^{n}\hat{R}_{k_{ij}}z_j + \mu_0\mu_1\cdots\mu_{k-1}\sum_{j=1}^{n}\hat{R}_{k_{ij}}L_j$$

$$\leq\ \delta_0\delta_1\cdots\delta_{k-1}\delta_k\,z_i\,||\varphi||^2_{L^2}e^{-\lambda\int_{t_0}^{t}h(s)ds} + \mu_0\mu_1\cdots\mu_{k-1}\mu_k L_i,\quad i\in\mathcal{N}. \tag{37}$$

This, together with (36) and $\delta_k \geq 1, \mu_k \geq 1$, leads to

$$\mathbb{E}V_i(x(t)) \leq\ \delta_0\cdots\delta_k\,z_i\,||\varphi||^2_{L^2}e^{-\lambda\int_{t_0}^{t}h(s)ds} + \mu_0\cdots\mu_k L_i,\ t\in[t_k-\tau,t_k]. \tag{38}$$

On the other hand, from $\mu_k \geq 1$ and (33)

$$\mathbb{E}\mathcal{L}V_i(x) \leq h(t)(p_{ii}\mathbb{E}V_i(x(t)) + \sum_{j=1}^{n} q_{ij}[\mathbb{E}V_j(x(t))]_\tau$$
$$+\mu_0\cdots\mu_k\hat{J}_i),\ \ t\in[t_k,t_{k+1}),\ i\in\mathcal{N},\ k\in\mathbb{N}. \tag{39}$$

By Lemma 1 again, the vector $\delta_0\cdots\delta_{k-1}\delta_k||\varphi||^2_{L^2}\,z \in \Omega_M(S)$. It follows from (38), the above inequality and Theorem 2 that

$$\mathbb{E}V_i(x(t)) \leq\ \delta_0\cdots\delta_k\,z_i\,||\varphi||^2_{L^2}e^{-\lambda\int_{t_0}^{t}h(s)ds}$$
$$+\mu_0\cdots\mu_k L_i,\ \ t\in[t_k,t_{k+1}),\ i\in\mathcal{N},\ k\in\mathbb{N}. \tag{40}$$

By the mathematical induction and (27), we conclude that

$$\mathbb{E}V_i(x(t) \leq\ e^{\delta(t_k-t_0)}\,z_i\,||\varphi||^2_{L^2}e^{-\lambda\int_{t_0}^{t}h(s)ds}$$
$$+e^{\mu}L_i,\ \ t\in[t_k,t_{k+1}),\ i\in\mathcal{N},\ k\in\mathbb{N}. \tag{41}$$

So, (30) hold. The proof is completed.

By Theorem 3, the following Corollary 1 and Corollary 2 are true.

Corollary 1. *If* (A_1) - (A_5) *hold and* $\lim\limits_{t\to\infty}\int_{t_0}^{t}[\lambda h(s)-\delta]ds=\infty$, *then the neural networks* (1) *are dissipative in mean square.*

Corollary 2. *If* (A_1) - (A_5) *hold,* $h(t)\equiv h$ *for* $t\geq t_0$, h *is positive constant satisfying* $\lambda h-\delta>0$ *then the neural networks* (1) *are exponentially dissipative in mean square.*

Corollary 3. *If* (A_1) - (A_5) *hold and* $h(t)$ *is a continuous* ω*-periodic function satisfying* $\frac{\lambda\rho}{\omega}-\delta>0$, *where* $\omega>0$ *and* $\rho=\int_0^\omega h(s)ds$, *then the neural networks* (1) *are exponentially dissipative in mean square.*

Proof. Since $h(t)$ is a continuous ω-periodic function and $h(t) \geq 0$, we have

$$\int_{t_0}^{t} h(s)ds \geq (\frac{t-t_0}{\omega} - 1)\int_0^{\omega} h(s)ds = (\frac{t-t_0}{\omega} - 1)\rho, \quad \forall\, t \geq t_0. \tag{42}$$

By (42) and Theorem 3, Corollary 3 holds.

Remark 3. It is evident that neural networks (1) have an equilibrium solution $x(t) \equiv 0$ from (A_1), (A_2) and (A_4) when $\hat{J} = 0$.

Corollary 4. *Suppose that $\hat{J} = 0$ and Conditions of Corollary 2 (or Corollary 3) hold. Then the zero solution of* (1) *is exponentially stable in mean square.*

Proof. From $\hat{J} = 0$, we can obtain $L = 0$ in (28). So Corollary 4 holds by (30) and Corollary 2 (or Corollary 3).

5 Example

Example 1. Consider the following impulsive stochastic neural networks:

$$\begin{cases} dx_1(t) = |\sin t|[-6x_1(t) + x_1(t-1) + x_2(t-\frac{1}{2})]dt \\ \qquad\qquad +\sqrt{6|\sin t|}x_1(t)dw_1(t) + \sqrt{|\sin t|}dw_2(t), \quad t \neq t_k, \\ dx_2(t) = |\sin t|[-5x_2(t) + x_1(t-\frac{1}{3}) + x_2(t-\frac{1}{4})]dt \\ \qquad\qquad +\sqrt{|\sin t|}dw_1(t) - \sqrt{|\sin t|}x_2(t-1)dw_2(t), \quad t \neq t_k, \\ x_1(t_k) = \frac{1}{4}e^{\frac{1}{4^k}}x_1(t_k^-) - \frac{1}{8}e^{\frac{1}{4^k}}x_2(t_k^-), \\ x_2(t_k) = -\frac{3}{8}e^{\frac{1}{4^k}}x_1(t_k^-) + \frac{1}{8}e^{\frac{1}{4^k}}x_2(t_k^-), \end{cases} \tag{43}$$

where $t_k = 2k$, $k \in \mathbb{N}$. So, we can choose the parameters in (A_1), (A_2), (A_4), (A_5) by taking $\tau = 1$, $h(t) = |\sin t|$, $H = 1$, $\hat{a}_1 = 6$, $\hat{a}_2 = 5$, $\hat{b}_{11} = \hat{b}_{12} = \hat{b}_{21} = \hat{b}_{22} = v_1 = v_2 = 1$, $\tilde{J}_1 = \tilde{J}_2 = 0$, $c_1 = 6$, $c_2 = 0$, $d_1 = 0$, $d_2 = 1$, $\bar{J}_1 = \bar{J}_2 = 1$,

$$R_k = \frac{1}{8}e^{\frac{1}{4^k}}\begin{pmatrix} 2 & 1 \\ 3 & 1 \end{pmatrix}, \quad \hat{R}_k = \frac{1}{64}e^{\frac{2}{4^k}}\begin{pmatrix} 6 & 3 \\ 12 & 4 \end{pmatrix}. \tag{44}$$

Then

$$P = \begin{pmatrix} -4 & 0 \\ 0 & -8 \end{pmatrix}, \quad Q = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}, \quad \hat{J} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad L = \begin{pmatrix} \frac{7}{17} \\ \frac{4}{17} \end{pmatrix}. \tag{45}$$

It is easy to prove that $S = -(P + Q)$ is an M-matrix and

$$\Omega_M(S) = \left\{ (z_1, z_2)^T > 0 \mid \frac{1}{6}z_1 < z_2 < 3z_1 \right\}. \tag{46}$$

Let $z = (1, 1)^T \in \Omega_M(S)$, $\lambda = 0.5$, $\delta = 0.25$, $\delta_k = \mu_k = e^{\frac{2}{4^k}} \geq 1$, $k \in \mathbb{N}$, then

$$\frac{\ln \delta_k}{t_k - t_{k-1}} \leq \frac{\ln e^{\frac{2}{4^k}}}{2} = \frac{1}{4^k} \leq 0.25 = \delta, \ \mu = \sum_{k=1}^{\infty} \ln \mu_k = \sum_{k=1}^{\infty} \frac{2}{4^k} = \frac{2}{3} < \infty, \tag{47}$$

$$[\lambda I + P + Qe^{\lambda H}]z < 0, \quad \hat{R}_k z \leq \delta_k z, \quad \hat{R}_k L \leq \mu_k L. \tag{48}$$

So, conditions (A_1) - (A_5) are satisfied. Furthermore, $h(t) = |\sin t|$ is a continuous π-periodic nonnegative function satisfying $\frac{\lambda\rho}{\pi} - \delta > 0$, where $\rho = \int_0^\pi h(s)ds = 2$. By Corollary 3, the neural networks (43) are exponentially dissipative in mean square.

Acknowledgements. The work was supported by National Natural Science Foundation of China under the grant No. 10926033, 10971147 and 10971240, A Project Supported by Scientific Reserch Fund of SiChuan Provincial Education Department (08zb026), Key Research Project of Sichuan Normal University, Natural Science Foundation of Chongqing under Grant CSTC2008BB2364.

References

1. Zhao, H., Ding, N.: Dynamic Analysis of Stochastic Bidirectional Associative Memory Neural Networks with Delays. Chaos, Solitons & Fractals 32, 1692–1702 (2007)
2. Xu, D., Yang, Z.: Impulsive Delay Differential Inequality and Stability of Neural Networks. J. Math. Anal. Appl. 305, 107–120 (2005)
3. Wang, X., Guo, Q., Xu, D.: Exponential P-stability of Impulsive Stochastic Cohen-Grossberg Neural Networks with Mixed Delays. Mathematics and Computers in Simulation 79, 1698–1710 (2009)
4. Xu, L., Xu, D.: Exponential P-stability of Impulsive Stochastic Neural Networks with Mixed Delays. Chaos, Solitons & Fractals 41, 263–272 (2009)
5. Yang, Z., Xu, D., Xiang, L.: Exponential P-stability of Impulsive Stochastic Differential Equations with Delays. Phys. Lett. A 359, 129–137 (2006)
6. Huang, Y., Xu, D., Yang, Z.: Dissipativity and Periodic Attractor for Non-autonomous Neural Networks with Time-varying Delays. Neurocomputing 70, 2953–2958 (2007)

Stability Analysis of Discrete Hopfield Neural Networks Combined with Small Ones

Weigen Wu, Jimin Yuan, Jun Li, Qianrong Tan, and Xing Yin

School of Computer Science and Technology, Pan Zhi Hua University,
Panzhihua 637000, China
wwg@mail.pzhu.edu.cn

Abstract. The stability of Discrete Hopfield neural networks (DHNNs) is very important in various applications, but stability analysis of complex DHNNs is difficult. However, stability analysis of simple and small DHNNs is easy to obtain. In this paper, we study on the stability of DHNNs combined with two small ones that maybe have partially or completely different neurons, and consider the connected weights of different neurons. And some new stability conditions of the DHNNs are obtained by studying the stability of small ones. Those results provide the guided significance for designing a DHNNs and afford benefit for stability analysis of DHNNs. The obtained results provide some theory bases of the application of DHNNs.

1 Introduction

Discrete Hopfield neural networks(DHNNs) is one of the famous neural networks with a wide range of applications, such as content addressable memory, pattern recognition, and combinatorial optimization. Because the stability of DHNNs is the foundation of the network's applications, the stability analysis of the DHNNs has attracted considerable interest. Most articles only discuss symmetric、anti-symmetric 、positive and diagonally dominant of connected matrixes[1-10]. Xu Z B proved that the networks will be globally convergent to a stable state if the interconnection matrix is weekly diagonally dominant. Furthermore, under one of conditions assuring global convergence of the network, the maximal attraction radius is found to be half of the distribution distance of the state to the network[1,2,3,9]. But most of methods of the stability analysis of the DHNNs seldomly consider that a DHNNs may be combined with two or more DHNNses.

When we analyze the stability of complex DHNNs $N=(W,\theta)$ is difficult, we may decompose the weight matrix of the complex DHNNs into many matrixes of the simple DHNNses, for example: $N=(W,\theta)$ is decomposed into two DHNNses $N^1=(A,\alpha)$ and $N^2=(B,\beta)$, and N^1 and N^2 have the same neuron, that is $\theta=\alpha+\beta$ and $W=A+B$, where Ma R N. thinks that B is an increment of the weight matrix A and a new weight matrix W is obtained[3,9,10]. However, neurons of N^1 are partially or completely different from neurons of N^2 in reality. We know that arbitrary DHNNs may consist of some simple DHNNses. So, in this paper it is

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 745–752, 2010.

the main aim that the stability of the DHHNs is obtained by researching some simple DHNNses, which maybe have partially or completely different neurons, when they are combined.

This paper has the following organization. In this section (Section 1), we provide the introduction. In Section 2, we analyze the stability of DHNNs combined with two and more ones. In Section 3 offers the conclusions of this paper.

2 Main Results

With the previous results, we mainly consider stability analysis of DHNNs combined with two completely and partially different DHNNses.

2.1 Stability Analysis of DHNNs Combined with Two Completely Different DHNNses

In this section, we know that neurons of N^1 are the completely same to neurons of N^2 and the number of neuron is the same from Ma Runnian method. It is not good that neurons of N^1 are the completely same to neurons of N^2, for it is unpractical. In really, neurons of N^1 are the partially or completely different from neurons of N^2.

The weight matrix connected between N^1 and N^2 is ignored in example 1. We consider the weight matrix W_A (W_B) from neurons of N^1 (N^2) to neurons of N^2 (N^1). So, the weight matrix W of N is

$$\begin{bmatrix} A & W_A \\ W_B & B \end{bmatrix}$$

and we obtain:

Theorem 1. *Let* $N^1 = (A, \alpha), A = \{a_{ij}\}_{n\times n}$ *and* $N^2 = (B, \beta), B = \{b_{ij}\}_{m\times m}$

be two DHNNses. Neurons of N^1 *are completely different from neurons of* N^2 *(that is* $N^1 \bigcap N^2 = \varnothing$*).* $N = N^1 \odot N^2 = (W, \theta)$*, where*

$$W = A \odot B = \begin{bmatrix} A & W_A \\ W_B & B \end{bmatrix} \text{ and } \theta = \alpha \odot \beta = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \; W_A = \{w_{ij}^a\}_{n\times m} \; (\; W_B = \{w_{ij}^b\}_{m\times n} \;)$$

is the weight matrix from neurons of N^1 (N^2) *to neurons of* N^2 (N^1). $\odot$ *is the combination of two completely different DHNNses. When the changes of energy of* N^1 *and* N^2 *are nonnegative that means* $\Delta E_{N^1}(t) \le 0$ *and* $\Delta E_{N^2}(t) \le 0$*, we have:*

1. *If* $W_A = 0$ *and* $W_B = 0$*, then* N *converges to a stable state;*
2. *If* W_A *is the anti-symmetric and transposed matrix of* W_B ($W_A + W_B^T = 0$)*, then* N *converges to a stable state.*

Proof: consider the energy function:

$$E(t) = -\tfrac{1}{2} X^T(t) W X(t) - X^T(t)\theta$$

$$= -\tfrac{1}{2} X^T(t) \begin{bmatrix} A & W_A \\ W_B & B \end{bmatrix} X(t) - X^T(t) \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

$$= -\tfrac{1}{2} \left(X_{N^1}^T(t), X_{N^2}^T(t) \right) \begin{bmatrix} A & W_A \\ W_B & B \end{bmatrix} \begin{pmatrix} X_{N^1}(t) \\ X_{N^2}(t) \end{pmatrix} - \left(X_{N^1}^T(t), X_{N^2}^T(t) \right) \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

$X_{N^1}(t)$ and $X_{N^2}(t)$ respectively indicate the state of N^1 and N^2.

$$E(t) = -\tfrac{1}{2} X_{N^1}^T(t) A X_{N^1}(t) - X_{N^1}^T(t)\alpha - \tfrac{1}{2} X_{N^2}^T(t) B X_{N^2}(t) - X_{N^2}^T(t)\beta$$
$$-\tfrac{1}{2} X_{N^1}^T(t) W_A X_{N^2}(t) - \tfrac{1}{2} X_{N^2}^T(t) W_B X_{N^1}(t)$$

If $W_A = 0$ and $W_B = 0$, or $W_A + W_B^T = 0$, then

$$E(t) = -\tfrac{1}{2} X_{N^1}^T(t) A X_{N^1}(t) - X_{N^1}^T(t)\alpha - \tfrac{1}{2} X_{N^2}^T(t) B X_{N^2}(t) - X_{N^2}^T(t)\beta$$

So,

$$\Delta E(t) = \Delta E_{N^1}(t) + \Delta E_{N^2}(t)$$

Because $\Delta E_{N^1}(t) \le 0$ and $\Delta E_{N^2}(t) \le 0$, $\Delta E(t) \le 0$ that N is stable. The proof is completed.

This theorem gives a method of judgment the stability of a DHNNs and provide the guided significance for designing a DHNNs.

2.2 Stability Analysis of DHNNs Combined with Two Partially Different DHNNses

In this section, we introduce a stability condition of DHNNs combined with N^1 and N^2 that neurons of N^1 are completely different from neurons of N^2 in theorem 1. However, how to work on DHNNs combined with N^1 and N^2 that neurons of N^1 are partially different from neurons of N^2, when two DHNNses are connected.

In order to analyze, neurons of N^1 are divided into two sets ($N^{11} = (A_1, \alpha_1)$ and $N^{12} = (A_4, \alpha_2)$). $A_1 = \{a_{ij}^1\}_{n_1 \times n_1}$ is the weight matrix of N^{11}, which the number of neuron is n_1. $A_4 = \{a_{ij}^4\}_{n_2 \times n_2}$ is the weight matrix of N^{12}, which the number of neuron is n_2. $A_2 = \{a_{ij}^2\}_{n_1 \times n_2}$ ($A_3 = \{a_{ij}^3\}_{n_2 \times n_1}$) is the weight matrix from neurons of N^{11} (N^{12}) to neurons of N^{12} (N^{11}). N^1 is denoted by $N^1 = (A, \alpha)$, where

$$A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \text{ and } \alpha = (\alpha_1, \alpha_2) \tag{1}$$

Similarly, neurons of N^2 are divided into two sets ($N^{21} = (B_1, \beta_1)$ and $N^{22} = (B_4, \beta_2)$). $B_1 = \{b_{ij}^1\}_{m_1 \times m_1}$ is the weight matrix of N^{21}, which the number of neuron is m_1. $B_4 = \{b_{ij}^4\}_{m_2 \times m_2}$ is the weight matrix of N^{22}, which the number of neuron

is m_2. $B_2 = \{b_{ij}^2\}_{m_1 \times m_2}$ ($B_3 = \{b_{ij}^3\}_{m_2 \times m_1}$) is the weight matrix from neurons of N^{21} (N^{22}) to neurons of N^{22} (N^{21}). N^2 is denoted by $N^2 = (B, \beta)$, where

$$B = \begin{bmatrix} B_4 & B_3 \\ B_2 & B_1 \end{bmatrix} \text{ and } \beta = (\beta_2, \beta_1) \tag{2}$$

We suppose that neurons of N^{12} are completely same to neurons of N^{22}, then $n_2 = m_2$. When N^1 is combined with N^2, $N = (W, \theta)$, where

$$W = A \circ B = \begin{bmatrix} A_1 & A_2 & W_A \\ A_3 & L & B_3 \\ W_B & B_2 & B_1 \end{bmatrix} \text{ and } \theta = \alpha \circ \beta = \begin{pmatrix} \alpha_1 \\ \gamma \\ \beta_1 \end{pmatrix} \tag{3}$$

L is the weight matrix of N^{12} or N^{22} ($L = A_4 + B_4$) , γ is the threshold value of public neurons of N^{12} and N^{22} ($\gamma = \alpha_2 + \beta_2$). So, N^L is denoted by $N^L = (L, \gamma)$. $W_A = \{w_{ij}^a\}_{n_1 \times m_1}$ ($W_B = \{w_{ij}^b\}_{m_1 \times n_1}$) is the weighted matrix from neurons of N^{11} (N^{21}) to neurons of N^{21} (N^{11}). In this paper, $\circ$ means the combination of two partially same DHNNses. We have:

Theorem 2: *Let $N^1 = (A, \alpha)$, which satisfies (1) , and $N^2 = (B, \beta)$, which satisfies (2), be two DHNNses. Neurons of N^1 are partially different from neurons of N^2. $N = N^1 \circ N^2 = (W, \theta)$, which satisfies (3).*

1) *If the changes of energy of N^1 and N^2 are nonnegative, and $W_A = W_B = 0 \vee W_A = -W_B^T$, then N converges to a stable state.*
2) *If the changes of energy of N^{11} , N^L and N^{21} are nonnegative, and $\left(A_2 = A_3 = 0 \vee A_2 = -A_3^T\right) \wedge \left(B_2 = B_3 = 0 \vee B_2 = -B_3^T\right) \wedge \left(W_A = W_B = 0 \vee W_A = -W_B^T\right)$, then N converges to a stable state.*

Proof : Consider the energy function:

$$E(t) = -\tfrac{1}{2} X^T(t) W X(t) - X^T(t) \theta$$

$$= -\tfrac{1}{2} \left(X_{N^{11}}^T(t), X_{N^L}^T(t), X_{N^{21}}^T(t) \right) \begin{bmatrix} A_1 & A_2 & W_A \\ A_3 & L & B_3 \\ W_B & B_2 & B_1 \end{bmatrix} \begin{pmatrix} X_{N^{11}}(t) \\ X_{N^L}(t) \\ X_{N^{21}}(t) \end{pmatrix}$$

$$- \left(X_{N^{11}}^T(t), X_{N^L}^T(t), X_{N^{21}}^T(t) \right) \begin{pmatrix} \alpha_1 \\ \gamma \\ \beta_1 \end{pmatrix}$$

$X_{N^{11}}(t)$, $X_{N^L}(t)$ and $X_{N^{21}}(t)$ respectively indicate the state of N^{11}, N^L and N^{21}.

Because $L = A_2 + B_2$ and $\gamma = \alpha_2 + \beta_2$

$$E(t) = -\tfrac{1}{2}\Big(X^T_{N^{11}}(t) A_1 X_{N^{11}}(t) + X^T_{N^L}(t) A_3 X_{N^{11}}(t) + X^T_{N^{11}}(t) A_2 X_{N^L}(t) +$$

$$X^T_{N^L}(t) A_4 X_{N^L}(t)\Big) - \Big(X^T_{N^{11}}(t)\alpha_1 + X^T_{N^L}(t)\alpha_2\Big)$$

$$-\tfrac{1}{2} X^T_{N^{21}}(t) W_B X_{N^{11}}(t) - \tfrac{1}{2} X^T_{N^{11}}(t) W_A X_{N^{21}}(t)$$

$$-\tfrac{1}{2}\Big(X^T_{N^{21}}(t) B_2 X_{N^L}(t) + X^T_{N^L}(t) B_3 X_{N^{21}}(t) + X^T_{N^{21}}(t) B_1 X_{N^{21}}(t) +$$

$$X^T_{N^L}(t) B_4 X_{N^L}(t)\Big) - \Big(X^T_{N^{21}}(t)\beta_1 + X^T_{N^L}(t)\beta_2\Big)$$

Part one: If $W_A = W_B = 0 \vee W_A = -W_B^T$, we obtain

$$E(t) = -\tfrac{1}{2}\Big(X^T_{N^{11}}(t), X^T_{N^L}(t)\Big)\begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}\begin{pmatrix} X_{N^{11}}(t) \\ X^T_{N^L}(t) \end{pmatrix} - \Big(X^T_{N^{11}}(t), X^T_{N^L}(t)\Big)\begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix}$$

$$-\tfrac{1}{2}\Big(X^T_{N^{21}}(t), X^T_{N^L}(t)\Big)\begin{bmatrix} B_1 & B_2 \\ B_3 & B_4 \end{bmatrix}\begin{pmatrix} X_{N^{21}}(t) \\ X^T_{N^L}(t) \end{pmatrix} - \Big(X^T_{N^{21}}(t), X^T_{N^L}(t)\Big)\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}$$

$$= -\tfrac{1}{2} X^T_{N^1}(t) A X_{N^1}(t) - X^T_{N^1}(t)\alpha - \tfrac{1}{2} X^T_{N^2}(t) B X_{N^2}(t) - X^T_{N^2}(t)\beta$$

So, $\Delta E(t) = \Delta E_{N^1}(t) + \Delta E_{N^2}(t)$

Because the changes of energy of N^1 and N^2 are nonnegative that N converges to a stable state.

Part two: If $\big(A_2 = A_3 = 0 \vee A_2 = -A_3^T\big) \wedge \big(B_2 = B_3 = 0 \vee B_2 = -B_3^T\big) \wedge \big(W_A = W_B = 0 \vee W_A = -W_B^T\big)$, we obtain:

$$E(t) = -\tfrac{1}{2} X^T_{N^{11}}(t) A_1 X_{N^{11}}(t) - X^T_{N^{11}}(t)\alpha_1 - \tfrac{1}{2} X^T_{N^{21}}(t) B_1 X_{N^{21}}(t) - X^T_{N^{21}}(t)\beta_1$$
$$-\tfrac{1}{2} X^T_{N^L}(t)\big(A_4 + B_4\big) X_{N^L}(t) - X^T_{N^L}(t)\big(\alpha_2 + \beta_2\big)$$

So, $\Delta E(t) = \Delta E_{N^{11}}(t) + \Delta E_{N^{21}}(t) + \Delta E_{N^L}(t)$

Because the changes of energy of N^1 and N^2 are nonnegative that N converges to a stable state. The proof is completed.

2.3 Corollary

Now, if we want to obtain a stable DHNNs, we may construct a stable large DHNNs with some stable simple DHNNses. Like this, it is easy to establish a large and stable DHNNs. So, we have

Corollary : *Let* $N^k = (W^k, \theta^k), W^k = \{w_{ij}^k\}_{n_k \times n_k}$ *be a DHNNs,* n_k *is the number of neuron of DHNNs* N^k *(*$1 < k \le q$*).*

1) *When* $\forall k, h$*, neurons of* N^k *are completely different from neurons of* N^h *and there are anti-symmetric or no connections between neurons of* N^k *and neurons of* N^h $(1 \le k, h \le q, k \ne h)$*, if* $\forall k$*, the change of energy of* N^k *is nonnegative, then* $N = \odot N^i \ (1 \le i \le q)$ *is stable too.*
2) *When* $\forall k, h$*, neurons of* N^k *are partially different from neurons of* N^h *and there are anti-symmetric or no connections between* N^{k1} *and* N^{h1} $(1 \le k, h \le q, k \ne h)$*, if* $\forall k, h,$ *, the changes of energy of* N^{k1}*,* $N^{L_{kh}}$ *and* N^{h1} *are nonnegative , then* $N = \circ N^i \ (1 \le i \le q)$ *is stable too.*
3) *When* $\forall k, h,$ *there are anti-symmetric or no connections between neurons of* N^k *that are different from* N^h *and neurons of* N^h *that are different from* N^k $(1 \le k, h \le q, k \ne h)$*, If* $\forall k$*, the change of energy of* N^k *is nonnegative, then* $N = \odot$ *or* $\circ N^i \ (1 \le i \le q)$ *is stable too.*

Proof : similarly to Theorem 1 and 2.

2.4 Examples

Example 1: Given a $N = (W, 0)$, where

$$W = \begin{bmatrix} 2 & -1 & -5 & 2 & 2 & 0 & 1 \\ 1 & 4 & -1 & -4 & -2 & 0 & 1 \\ 5 & -1 & 1 & 1 & 0 & 2 & -1 \\ -2 & -1 & 1 & 1 & -5 & 3 & 0 \\ -2 & 2 & 0 & 5 & 2 & -1 & 0 \\ 0 & 0 & -2 & -3 & 1 & 4 & -1 \\ -1 & -1 & 1 & 0 & 0 & 1 & 3 \end{bmatrix}$$

With the previous method we do not obtain the stability. However, let $N^{11} = (A_1, 0)$, where $A_1 = [2]$, $N^L = (L, 0)$, where

$$L = \begin{bmatrix} 4 & -1 & -4 \\ -1 & 1 & 1 \\ -1 & 1 & 1 \end{bmatrix}$$

$N^{21} = (B_1, 0)$, where

$$B_1 = \begin{bmatrix} 2 & -1 & 0 \\ 1 & 4 & -1 \\ 0 & 1 & 3 \end{bmatrix} \quad B_2 = \begin{bmatrix} 2 & 0 & 5 \\ 0 & -2 & -3 \\ -1 & 1 & 0 \end{bmatrix}$$

$A_2 = [-1 \quad -5 \quad 2]$, $A_3 = -A_2^T$, , $B_3 = -B_2^T$, $W_A = [2 \quad 0 \quad 1]$, and $W_B = [-2 \quad 0 \quad -1]^T$ respectively. With the Thermo 2 and $N^L = (L, 0)$ is stable[3], we obtain that $N = (W, 0)$, where

$$W = \begin{bmatrix} A_1 & A_2 & W_A \\ A_3 & L & B_3 \\ W_B & B_2 & B_1 \end{bmatrix}$$

is stable.

Example 2: In $N^1 = (A, \alpha)$, where

$$A = \begin{bmatrix} 3 & 1 & 0 & -8 \\ 1 & 2 & 2 & -1 \\ 0 & 2 & 4 & 0 \\ -8 & -1 & 0 & 6 \end{bmatrix}$$

and $\alpha = 0$. Here are four neurons that names are L1, L2, L3, and L4 respectively. In $N^2 = (B, \beta)$, where

$$B = \begin{bmatrix} 5 & 0 & 1 & 1 & 0 \\ -3 & 1 & 1 & -1 & 4 \\ 5 & 4 & 4 & -4 & 2 \\ 3 & -1 & -5 & 5 & 3 \\ 1 & -1 & 0 & -1 & 0 \end{bmatrix}$$

and $\beta = [2, 3, 0, 0, 4]^T$. There are five neurons that names are L2, L3, L4, L5 and L6, respectively. Then we know that L2, L3, and L4 are public neuron. If we $W_A = 0$ and $W_B = 0$, we obtain $N = (W, \theta)$, where

$$W = \begin{bmatrix} 3 & 1 & 0 & -8 & 0 & 0 \\ 1 & 7 & 2 & 0 & 1 & 0 \\ 0 & -1 & 5 & 1 & -1 & 4 \\ -8 & 4 & 4 & 10 & -4 & 2 \\ 0 & 3 & -1 & -5 & 5 & 3 \\ 0 & 1 & -1 & 0 & -1 & 0 \end{bmatrix}$$

and $\theta = [0,2,3,0,0,4]^T$. If N^1 combines with N^2 , then with the thermo 2 and $N^2 = (B, \beta)$ is stable[3] we have N is stable.

3 Conclusion

We mainly analyze the stability of DHNNs combined with two small ones. It is extend to the stability of DHNNs combined with many small ones that when $\forall k, h$, there are anti-symmetric or no connections between neurons of N^k that are different from N^h and neurons of N^h that are different from N^k $(1 \le k, h \le q, k \ne h)$, If $\forall k$, the energy of N^k is degressive, then $N = \odot \ or \ \circ N^i (1 \le i \le q)$ is stable in DHNNs set $\{N^1, N^2, \cdots, N^q\}$. Our results provide some theory bases of the application of DHNNs in combinatorial optimization and pattern recognition. In this paper, on the one hand, those results provide the guided significance for designing a DHNNs, on the other hand, afford benefit for stability analysis of DHNNs.

References

[1] Xu, Z.B., Kwong, C.P.: Global Convergence and Asymptotic Stability of Asymmetric Hopfield Neural Networks. J. Mathematical Analysis and Applications 191(3), 405–427 (1995)
[2] Lee, D.L.: New stability conditions for Hopfield neural networks in partial simultaneous update mode. IEEE Trans. on Neural Networks 10, 975–978 (1999)
[3] Ma, R.N., Zhang, Q., Xu, J.: Stability Study on the Discrete Hopfield Network in Various Updating Modes. Systems Engineering and Electronics 24(3), 91–94 (2002) (in Chinese)
[4] Sun, Y.: A generalized updating rule for modified Hopfield neural network for quadratic optimization. Neurocompting 19, 133–143 (1998)
[5] Xu, Z.B., Leung, Y., He, X.W.: Asymmetric bidirectional associative memories. IEEE Transactions on System Man Cybernetics 24, 1558–1564 (1994)
[6] Xu, Z.B., Hu, G.Q., Kwong, C.P.: Some efficient strategies for improving the eigenstructure method in synthesis of feedback neural networks. IEEE Transactions on Neural Networks 7, 233–245 (1996)
[7] Tsang Eric, C.C., Qiu, S.S., Yeung Daniel, S.: Stability analysis of a discrete Hopfield neural network with delay. Neurocomputing 70, 2598–2602 (2007)
[8] Zhang, Q., Wei, X.P., Xu, J.: On Global Exponential Stability of Discrete-Time Hopfield Neural Networks with Variable Delays. Discrete Dynamics in Nature and Society, Article ID 67675, 9 (2007)
[9] Xu, Z.B., Hu, G.Q., Kwong, C.P.: Asymmetric Hopfield-type Networks: Theory and Applications. Neural Networks 9(3), 483–501 (1996)
[10] Ma, R., Zhang, S., Lei, S.: Stability Conditions for Discrete Neural Networks in Partial Simultaneous Updating Mode. ISNN (1), 253–258 (2005)

Author Index