LNCS 6675

Derong Liu Huaguang Zhang
Marios Polycarpou Cesare Alippi
Haibo He (Eds.)

Advances in Neural Networks – ISNN 2011

8th International Symposium on Neural Networks, ISNN 2011
Guilin, China, May/June 2011
Proceedings, Part I

1 Part I

Springer

Lecture Notes in Computer Science 6675

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

Editorial Board

David Hutchison
Lancaster University, UK
Takeo Kanade
Carnegie Mellon University, Pittsburgh, PA, USA
Josef Kittler
University of Surrey, Guildford, UK
Jon M. Kleinberg
Cornell University, Ithaca, NY, USA
Alfred Kobsa
University of California, Irvine, CA, USA
Friedemann Mattern
ETH Zurich, Switzerland
John C. Mitchell
Stanford University, CA, USA
Moni Naor
Weizmann Institute of Science, Rehovot, Israel
Oscar Nierstrasz
University of Bern, Switzerland
C. Pandu Rangan
Indian Institute of Technology, Madras, India
Bernhard Steffen
TU Dortmund University, Germany
Madhu Sudan
Microsoft Research, Cambridge, MA, USA
Demetri Terzopoulos
University of California, Los Angeles, CA, USA
Doug Tygar
University of California, Berkeley, CA, USA
Gerhard Weikum
Max Planck Institute for Informatics, Saarbruecken, Germany

Derong Liu Huaguang Zhang
Marios Polycarpou Cesare Alippi
Haibo He (Eds.)

Advances in Neural Networks – ISNN 2011

8th International Symposium
on Neural Networks, ISNN 2011
Guilin, China, May 29 – June 1, 2011
Proceedings, Part I

Springer

Volume Editors

Derong Liu
Chinese Academy of Sciences, Institute of Automation
Key Laboratory of Complex Systems and Intelligence Science
Beijing 100190, China
E-mail: derong.liu@ia.ac.cn

Huaguang Zhang
Northeastern University, College of Information Science and Engineering
Shenyang, Liaoing 110004, China
E-mail: zhanghuaguang@ise.neu.edu.cn

Marios Polycarpou
University of Cyprus, Dept. of Electrical and Computer Engineering
75 Kallipoleos Avenue, 1678 Nicosia, Cyprus
E-mail: mpolycar@ucy.ac.cy

Cesare Alippi
Politecnico di Milano, Dip. di Elettronica e Informazione
Piazza L. da Vinci 32, 20133 Milano, Italy
E-mail: alippi@elet.polimi.it

Haibo He
University of Rhode Island
Dept. of Electrical, Computer and Biomedical Engineering
Kingston, RI 02881, USA
E-mail: he@ele.uri.edu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-21104-1 e-ISBN 978-3-642-21105-8
DOI 10.1007/978-3-642-21105-8
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011926887

CR Subject Classification (1998): F.1, F.2, D.1, G.2, I.2, C.2, I.4-5, J.1-4

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

© Springer-Verlag Berlin Heidelberg 2011
This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.
The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

ISNN 2011 – the 8th International Symposium on Neural Networks – was held in Guilin, China, as a sequel of ISNN 2004 (Dalian), ISNN 2005 (Chongqing), ISNN 2006 (Chengdu), ISNN 2007 (Nanjing), ISNN 2008 (Beijing), ISNN 2009 (Wuhan), and ISNN 2010 (Shanghai). ISNN has now become a well-established conference series on neural networks in the region and around the world, with growing popularity and increasing quality. Guilin is regarded as the most picturesque city in China. All participants of ISNN 2011 had a technically rewarding experience as well as memorable experiences in this great city.

ISNN 2011 aimed to provide a high-level international forum for scientists, engineers, and educators to present the state of the art of neural network research and applications in diverse fields. The symposium featured plenary lectures given by worldwide renowned scholars, regular sessions with broad coverage, and some special sessions focusing on popular topics.

The symposium received a total of 651 submissions from 1,181 authors in 30 countries and regions across all six continents. Based on rigorous reviews by the Program Committee members and reviewers, 215 high-quality papers were selected for publication in the symposium proceedings. We would like to express our sincere gratitude to all reviewers of ISNN 2011 for the time and effort they generously gave to the symposium. We are very grateful to the National Natural Science Foundation of China, the Institute of Automation of the Chinese Academy of Sciences, the Chinese University of Hong Kong, and the University of Illinois at Chicago for their financial support. We would also like to thank the publisher, Springer, for cooperation in publishing the proceedings in the prestigious series of *Lecture Notes in Computer Science.*

Guilin, May 2011

Derong Liu
Huaguang Zhang
Marios Polycarpou
Cesare Alippi
Haibo He

ISNN 2011 Organization

General Chairs

Marios M. Polycarpou	University of Cyprus, Cyprus
Paul J. Werbos	National Science Foundation, USA

Advisory Committee Chairs

Ruwei Dai	Chinese Academy of Sciences, China
Bo Zhang	Tsinghua University, China

Advisory Committee Members

Hojjat Adeli	Ohio State University, USA
Shun-ichi Amari	RIKEN Brain Science Institute, Japan
Dimitri P. Bertsekas	Massachusetts Institute of Technology, USA
Amit Bhaya	Federal University of Rio de Janeiro, Brazil
Tianyou Chai	Northeastern University, China
Guanrong Chen	City University of Hong Kong, Hong Kong
Andrzej Cichocki	RIKEN Brain Science Institute, Japan
Jay Farrell	University of California, Riverside, USA
Russell Eberhart	Indiana University-Purdue University, USA
David B. Fogel	Natural Selection, Inc., USA
Walter J. Freeman	University of California-Berkeley, USA
Kunihiko Fukushima	Kansai University, Japan
Marco Gilli	Politecnico di Torino, Italy
Aike Guo	Chinese Academy of Sciences, China
Zhenya He	Southeast University, China
Tom Heskes	Radboud University Nijmegen, The Netherlands
Janusz Kacprzyk	Polish Academy of Sciences, Poland
Nikola Kasabov	Auckland University of Technology, New Zealand
Okyay Kaynak	Bogazici University, Turkey
Anthony Kuh	University of Hawaii, USA
Deyi Li	National Natural Science Foundation of China, China
Yanda Li	Tsinghua University, China
Chin-Teng Lin	National Chiao Tung University, Taiwan
Robert J. Marks II	Baylor University, USA
Erkki Oja	Helsinki University of Technology, Finland
Nikhil R. Pal	Indian Statistical Institute, India
Jose C. Príncipe	University of Florida, USA

Leszek Rutkowski	Czestochowa University of Technology, Poland
Jennie Si	Arizona State University, USA
Youxian Sun	Zhejiang University, China
DeLiang Wang	Ohio State University, USA
Fei-Yue Wang	Chinese Academy of Sciences, China
Shoujue Wang	Chinese Academy of Sciences, China
Zidong Wang	Brunel University, UK
Cheng Wu	Tsinghua University, Beijing, China
Donald Wunsch II	Missouri University of Science and Technology, USA
Lei Xu	The Chinese University of Hong Kong, Hong Kong
Shuzi Yang	Huazhong University of Science and Technology, China
Xin Yao	University of Birmingham, UK
Gary G. Yen	Oklahoma State University, USA
Nanning Zheng	Xi'An Jiaotong University, China
Jacek M. Zurada	University of Louisville, USA

Steering Committee Chair

Jun Wang	Chinese University of Hong Kong, Hong Kong

Steering Committee Members

Jinde Cao	Southeast University, China
Shumin Fei	Southeast University, China
Min Han	Dalian University of Technology, China
Xiaofeng Liao	Chongqing University, China
Bao-Liang Lu	Shanghai Jiao Tong University, China
Yi Shen	Huazhong University of Science and Technology, China
Fuchun Sun	Tsinghua University, China
Hongwei Wang	Huazhong University of Science and Technology, China
Zongben Xu	Xi'An Jiaotong University, China
Zhang Yi	Sichuan University, China
Wen Yu	National Polytechnic Institute, Mexico

Organizing Committee Chairs

Derong Liu	Chinese Academy of Sciences, China
Huaguang Zhang	Northeastern University, China

Program Chairs

Cesare Alippi	Politecnico di Milano, Italy
Bhaskhar DasGupta	University of Illinois at Chicago, USA
Sanqing Hu	Hangzhou Dianzi University, China

Plenary Sessions Chairs

Frank L. Lewis	University of Texas at Arlington, USA
Changyin Sun	Southeast University, China

Special Sessions Chairs

Amir Hussain	University of Stirling, UK
Jinhu Lu	Chinese Academy of Sciences, China
Stefano Squartini	Università Politecnica delle Marche, Italy
Liang Zhao	University of Sao Paulo, Brazil

Finance Chairs

Hairong Dong	Beijing Jiaotong University, China
Cong Wang	South China University of Technology, China
Zhigang Zeng	Huazhong University of Science and Technology, China
Dongbin Zhao	Chinese Academy of Sciences, China

Publicity Chairs

Zeng-Guang Hou	Chinese Academy of Sciences, China
Manuel Roveri	Politecnico di Milano, Italy
Songyun Xie	Northwestern Polytechnical University, China
Nian Zhang	University of the District of Columbia, USA

European Liaisons

Danilo P. Mandic	Imperial College London, UK
Alessandro Sperduti	University of Padova, Italy

Publications Chairs

Haibo He	Stevens Institute of Technology, USA
Wenlian Lu	Fudan University, China
Yunong Zhang	Sun Yat-sen University, China

Registration Chairs

Xiaolin Hu	Tsinghua University, China
Zhigang Liu	Southwest Jiaotong University, China
Qinglai Wei	Chinese Academy of Sciences, China

Local Arrangements Chairs

Xuanju Dang	Guilin University of Electronic Technology, China
Xiaofeng Lin	Guangxi University, China
Yong Xu	Guilin University of Electronic Technology, China

Electronic Review Chair

Tao Xiang	Chongqing University, China

Symposium Secretariat

Ding Wang	Chinese Academy of Sciences, China

ISNN 2011 International Program Committee

Jose Aguilar	Universidad de los Andes, Venezuela
Haydar Akca	United Arab Emirates University, UAE
Angelo Alessandri	University of Genoa, Italy
Luís Alexandre	Universidade da Beira Interior, Portugal
Bruno Apolloni	University of Milan, Italy
Marco Antonio Moreno Armendáriz	Instituto Politecnico Nacional, Mexico
K. Vijayan Asari	Old Dominion University, USA
Amir Atiya	Cairo University, Egypt
Monica Bianchini	Università degli Studi di Siena, Italy
Salim Bouzerdoum	University of Wollongong, Australia
Ivo Bukovsky	Czech Technical University, Czech Republic
Xindi Cai	APC St. Louis, USA
Jianting Cao	Saitama Institute of Technology, Japan
M. Emre Celebi	Louisiana State University, USA
Jonathan Hoyin Chan	King Mongkut's University of Technology, Thailand
Ke Chen	University of Manchester, UK
Songcan Chen	Nanjing University of Aeronautics and Astronautics, China
YangQuan Chen	Utah State University, USA
Yen-Wei Chen	Ritsumeikan University, Japan
Zengqiang Chen	Nankai University, China

Jianlin Cheng	University of Missouri Columbia, USA
Li Cheng	NICTA Australian National University, Australia
Long Cheng	Chinese Academy of Sciences, China
Xiaochun Cheng	University of Reading, UK
Sung-Bae Cho	Yonsei University, Korea
Pau-Choo Chung	National Cheng Kung University, Taiwan
Jose Alfredo Ferreira Costa	Federal University, UFRN, Brazil
Sergio Cruces-Alvarez	University of Seville, Spain
Lili Cui	Northeastern University, China
Chuangyin Dang	City University of Hong Kong, Hong Kong
Xuanju Dang	Guilin University of Electronic Technology, China
Mingcong Deng	Okayama University, Japan
Ming Dong	Wayne State University, USA
Gerard Dreyfus	ESPCI-ParisTech, France
Haibin Duan	Beihang University, China
Wlodzislaw Duch	Nicolaus Copernicus University, Poland
El-Sayed El-Alfy	King Fahd University of Petroleum and Minerals, Saudi Arabia
Pablo Estevez	Universidad de Chile, Chile
Jay Farrell	University of California Riverside, USA
Wai-Keung Fung	University of Manitoba, Canada
John Gan	University of Essex, UK
Junbin Gao	Charles Sturt University, Australia
Xiao-Zhi Gao	Helsinki University of Technology, Finland
Anya Getman	University of Nevada Reno, USA
Xinping Guan	Shanghai Jiao Tong University, China
Chengan Guo	Dalian University of Technology, China
Lejiang Guo	Huazhong University of Science and Technology, China
Ping Guo	Beijing Normal University, China
Qing-Long Han	Central Queensland University, Australia
Haibo He	Stevens Institute of Technology, USA
Zhaoshui He	RIKEN Brain Science Institute, Japan
Tom Heskes	Radboud University Nijmegen, The Netherlands
Zeng-Guang Hou	Chinese Academy of Sciences, China
Zhongsheng Hou	Beijing Jiaotong University, China
Chun-Fei Hsu	Chung Hua University, Taiwan
Huosheng Hu	University of Essex, UK
Jinglu Hu	Waseda University, Japan
Guang-Bin Huang	Nanyang Technological University, Singapore
Ting Huang	University of Illinois at Chicago, USA
Tingwen Huang	Texas A&M University at Qatar
Marc Van Hulle	Katholieke Universiteit Leuven, Belgium
Amir Hussain	University of Stirling, UK
Giacomo Indiveri	ETH Zurich, Switzerland

Danchi Jiang	University of Tasmania, Australia
Haijun Jiang	Xinjiang University, China
Ning Jin	University of Illinois at Chicago, USA
Yaochu Jin	Honda Research Institute Europe, Germany
Joarder Kamruzzaman	Monash University, Australia
Qi Kang	Tongji University, China
Nikola Kasabov	Auckland University, New Zealand
Yunquan Ke	Shaoxing University, China
Rhee Man Kil	Korea Advanced Institute of Science and Technology, Korea
Kwang-Baek Kim	Silla University, Korea
Sungshin Kim	Pusan National University, Korea
Arto Klami	Helsinki University of Technology, Finland
Leo Li-Wei Ko	National Chiao Tung University, Taiwan
Mario Koeppen	Kyuhsu Institute of Technology, Japan
Stefanos Kollias	National Technical University of Athens, Greece
Sibel Senan Kucur	Istanbul University, Turkey
H.K. Kwan	University of Windsor, Canada
James Kwok	Hong Kong University of Science and Technology, Hong Kong
Edmund M.K. Lai	Massey University, New Zealand
Chuandong Li	Chongqing University, China
Kang Li	Queen's University Belfast, UK
Li Li	Tsinghua University, China
Michael Li	Central Queensland University, Australia
Shaoyuan Li	Shanghai Jiao Tong University, China
Shutao Li	Hunan University, China
Xiaoou Li	CINVESTAV-IPN, Mexico
Yangmin Li	University of Macao, Macao
Yuanqing Li	South China University of Technology, China
Hualou Liang	University of Texas at Houston, USA
Jinling Liang	Southeast University, China
Yanchun Liang	Jilin University, China
Lizhi Liao	Hong Kong Baptist University
Alan Wee-Chung Liew	Griffith University, Australia
Aristidis Likas	University of Ioannina, Greece
Chih-Jen Lin	National Taiwan University, Taiwan
Ju Liu	Shandong University, China
Meiqin Liu	Zhejiang University, China
Yan Liu	Motorola Labs, Motorola, Inc., USA
Zhenwei Liu	Northeastern University, China
Bao-Liang Lu	Shanghai Jiao Tong University, China
Hongtao Lu	Shanghai Jiao Tong University, China
Jinhu Lu	Chinese Academy of Sciences, China
Wenlian Lu	Fudan University, China

Yanhong Luo	Northeastern University, China
Jinwen Ma	Peking University, China
Malik Magdon-Ismail	Rensselaer Polytechnic Institute, USA
Danilo Mandic	Imperial College London, UK
Francesco Marcelloni	University of Pisa, Italy
Francesco Masulli	Università di Genova, Italy
Matteo Matteucci	Politecnico di Milano, Italy
Patricia Melin	Tijuana Institute of Technology, Mexico
Dan Meng	Southwest University of Finance and Economics, China
Yan Meng	Stevens Institute of Technology, USA
Valeri Mladenov	Technical University of Sofia, Bulgaria
Roman Neruda	Academy of Sciences of the Czech Republic, Czech Republic
Ikuko Nishikawa	Ritsumei University, Japan
Erkki Oja	Aalto University, Finland
Seiichi Ozawa	Kobe University, Japan
Guenther Palm	Universität Ulm, Germany
Christos Panayiotou	University of Cyprus, Cyprus
Shaoning Pang	Auckland University of Technology, New Zealand
Thomas Parisini	University of Trieste, Italy
Constantinos Pattichis	University of Cyprus, Cyprus
Jaakko Peltonen	Helsinki University of Technology, Finland
Vincenzo Piuri	University of Milan, Italy
Junfei Qiao	Beijing University of Technology, China
Manuel Roveri	Politecnico di Milano, Italy
George Rovithakis	Aristole University of Thessaloniki, Greece
Leszek Rutkowski	Technical University of Czestochowa, Poland
Tomasz Rutkowski	RIKEN Brain Science Institute, Japan
Sattar B. Sadkhan	University of Babylon, Iraq
Toshimichi Saito	Hosei University, Japan
Karl Sammut	Flinders University, Australia
Edgar Sanchez	CINVESTAV, Mexico
Marcello Sanguineti	University of Genoa, Italy
Gerald Schaefer	Aston University, UK
Furao Shen	Nanjing University, China
Daming Shi	Nanyang Technological University, Singapore
Hideaki Shimazaki	RIKEN Brain Science Institute, Japan
Qiankun Song	Chongqing Jiaotong University, China
Ruizhuo Song	Northeastern University, China
Alessandro Sperduti	University of Padua, Italy
Stefano Squartini	Università Politecnica delle Marche, Italy
Dipti Srinivasan	National University of Singapore, Singapore
John Sum	National Chung Hsing University, Taiwan
Changyin Sun	Southeast University, China

Johan Suykens	Katholieke Universiteit Leuven, Belgium
Roberto Tagliaferri	University of Salerno, Italy
Norikazu Takahashi	Kyushu University, Japan
Ah-Hwee Tan	Nanyang Technological University, Singapore
Ying Tan	Peking University, China
Toshihisa Tanaka	Tokyo University of Agriculture and Technology, Japan
Hao Tang	Hefei University of Technology, China
Qing Tao	Chinese Academy of Sciences, China
Ruck Thawonmas	Ritsumeikan University, Japan
Sergios Theodoridis	University of Athens, Greece
Peter Tino	Birmingham University, UK
Christos Tjortjis	University of Manchester, UK
Ivor Tsang	Nanyang Technological University, Singapore
Masao Utiyama	National Institute of Information and Communications Technology, Japan
Marley Vellasco	PUC-Rio, Brazil
Alessandro E.P. Villa	Université de Lausanne, Switzerland
Draguna Vrabie	University of Texas at Arlington, USA
Bing Wang	University of Hull, UK
Dan Wang	Dalian Maritime University, China
Dianhui Wang	La Trobe University, Australia
Ding Wang	Chinese Academy of Sciences, China
Lei Wang	Australian National University, Australia
Lei Wang	Tongji University, China
Wenjia Wang	University of East Anglia, UK
Wenwu Wang	University of Surrey, USA
Yingchun Wang	Northeastern University, China
Yiwen Wang	Hong Kong University of Science and Technology, Hong Kong
Zhanshan Wang	Northeastern University, China
Zhuo Wang	University of Illinois at Chicago, USA
Zidong Wang	Brunel University, UK
Qinglai Wei	Chinese Academy of Sciences, China
Yimin Wen	Hunan Institute of Technology, China
Wei Wu	Dalian University of Technology, China
Cheng Xiang	National University of Singapore, Singapore
Degui Xiao	Hunan University, China
Songyun Xie	Northwestern Polytechnical University, China
Rui Xu	Missouri University of Science and Technology, USA
Xin Xu	National University of Defense Technology, China
Yong Xu	Guilin University of Electronic Technology, China
Jianqiang Yi	Chinese Academy of Sciences, China
Zhang Yi	Sichuan University, China

Dingli Yu	Liverpool John Moores University, UK
Xiao-Hua Yu	California Polytechnic State University, USA
Xiaoqin Zeng	Hohai University, China
Zhigang Zeng	Huazhong University of Science and Technology, China
Changshui Zhang	Tsinghua University, China
Huaguang Zhang	Northeastern University, China
Jianghai Zhang	Hangzhou Dianzi University, China
Jie Zhang	University of New Castle, UK
Kai Zhang	Lawrence Berkeley National Lab, USA
Lei Zhang	Sichuan University, China
Nian Zhang	University of the District of Columbia, USA
Xiaodong Zhang	Wright State University, USA
Xin Zhang	Northeastern University, China
Yunong Zhang	Sun Yat-sen University, China
Dongbin Zhao	Chinese Academy of Sciences, China
Hai Zhao	Shanghai Jiao Tong University, China
Liang Zhao	University of São Paulo, Brazil
Mingjun Zhong	University of Glasgow, UK
Weihang Zhu	Lamar University, USA
Rodolfo Zunino	University of Genoa, Italy

Table of Contents – Part I

Neurodynamics and Complex Systems

Table of Contents – Part II

Visual Perception and Pattern Recognition

Motion, Tracking and Object Recognition

Table of Contents – Part III

Adaptive and Hybrid Intelligent Systems

Evaluating Humans' Implicit Attitudes towards an Embodied Conversational Agent

Andrey Kiselev[1], Niyaz Abdikeev[2], and Toyoaki Nishida[1]

[1] Dept. of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University; Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501 Japan
andrewak@ii.ist.i.kyoto-u.ac.jp, nishida@i.kyoto-u.ac.jp
[2] Plekhanov Russian Academy of Economics; 6, Stremyanny lane, Moscow 116998 Russia
nabd@rea.ru

Abstract. This paper addresses the problem of evaluating embodied conversational agents in terms of their communicative performance. We show our attempt to evaluate humans' implicit attitudes towards different kinds of information presenting by embodied conversational agents using the Implicit Association Test (IAT) rather than gathering explicit data using interviewing methods. We conducted an experiment in which we use the method of indirect measurements with the IAT. The conventional procedure and scoring algorithm of the IAT were used in order to discover possible issues and solutions for future experiments. We discuss key differences between the conventional usage of the IAT and using the IAT in our experiment for evaluating embodied conversational agents using unfamiliar information as test data.

Keywords: Embodied Conversational Agent, Implicit Association Test.

1 Introduction

Discovering humans' attitudes has become an important research topic for the past several decades. A number of methods based on different psychological theories have been proposed and various experiments showing strong and weak points of each method have been conducted. Being required by a wide variety of applications, methods of discovering humans' attitudes have developed very intensively.

One of the fields where gathering attitudes is widely used is human-computer cooperative work. When humans deal with not only computers, but with anthropomorphic artifacts, they tend to treat computers as distinct social actors, using the same social norms and rules for computers as they use for humans [1]. This makes communication among humans and computers much more important than ever for achieving high efficiency in collaborative tasks.

In our work we assume that although humans can adapt to any kind of interface, the most effective is one given to us by nature – natural face-to-face communication with an Embodied Conversational Agent [2] which includes a rich set of verbal and non-verbal cues. Providing computers with the abilities to communicate with us in a natural way, we can dramatically increase the efficiency of human-computer collaborative performance.

Nevertheless we believe that introducing state-of-art technologies in graphics, speech processing and dialog management is not the only condition of success in this

© Springer-Verlag Berlin Heidelberg 2011

task. A number of issues related to social and cultural aspects of communications among humans and ECAs should be considered. Gathering feedback from humans about their attitudes towards agents is used in a number of research projects as a measure of team effectiveness. This is supported by the assumption that the more natural and pleasant interaction is with an agent, the more effective the performance of the collaborative task. In order to discover humans' attitudes a number of methods based on the self-report such as surveys and interviews are used. All of these methods have unquestionable advantage such as simplicity and low cost, however they all tend to rely upon a humans' awareness, honesty, cultural aspects.

Another method for investigating effects of ECAs is measuring subjects' performance in different kinds of collaborative tasks, such as a tutoring-learning task, where subjects have to cooperate with artificial social actors in order to achieve a goal. Normally a combination of statistically significant surveys with collaborative task performance measurement gives high validity, however other metrics are often needed in order to verify results.

The Implicit Association Test [3, 4] is a powerful psychological tool which has been already used for more than a decade in psychology. We believe that IAT can potentially be a powerful addendum to all other kinds of research tools in the AI field.

The IAT is a well-known and established tool, however further developments of the procedure and scoring algorithms of the IAT are ongoing, aiming to solve a number of different issues of the original IAT. One of the best known modifications of the test is so-called Go/No-go Association Task [5], which aims to eliminate the need to bring a pair of categories into comparison. There are also some other modifications of the test, such as a Brief IAT [6] and Single Attribute IAT [7], which aim to solve known issues, simplify and improve the original test.

This goal of our research is to investigate a modification of the IAT which will extend the application domain of the conventional test to the possibility of using it with unfamiliar information and for indirect attitude measurements.

In this paper we show our attempt to use the conventional IAT in order to evaluate humans' attitudes towards slightly different kinds of presenting information by the same social actor. However our application method differs in principle from the original test application, we used a conventional test without any modification in order to discover it's possible drawbacks and find solutions.

The paper is organized as follows. Section 2 shows key differences between conventional IAT and applying a test to assess different types of presenting information. Section 3 contains a description and results of a conducted experiment. Known issues and future work directions are discussed in Section 4. The paper is concluded in Section 5.

2 Hypothesis

The Implicit Association Test is a very powerful psychological tool which can be used in order to discover a subjects' implicit preferences towards different categories. Particularly, IAT can be used to measure attitudes towards different kinds of objects and concepts, stereotypes, self-esteem and self-identity. The test requires a subject's rapid (in fact, as fast as possible) categorization of stimuli which appear on a screen.

The problem is that the test will give a reliable result if and only if subjects make a reasonable number of mistakes, trying to keep a balance between rapid categorization without thinking and spontaneous key pressing. In order to achieve reliable results, subject are required to be a fluent English readers (if a test is conducted in English) and be aware of the topic of the test.

Some very well-known tests allows us to measure attitudes towards, for example, flowers and insects, different races, and food preferences. If flowers and insects are assessed, the stimuli are names or images of particular flowers and insects.

In our case we want to compare two methods of presenting information. We just want to know whether a virtual agent with rich animation can attract subjects more than just audio-presentation by the same voice as used in the case of agent. Noticeably, in this case we are talking about the same social actor. We do not compare different agents. We would like to go deeper and compare interfaces of the same agent.

In order to achieve this goal we use two slightly different information blocks, one of which is presented by a full-featured agent and the second by voice only. This causes the principal difference between our application and original usage of the IAT: in our case subjects are not familiar with these information blocks. These are not things which the subject uses every day. Thus, as opposed to the original IAT, in our test subject are expected to make not only misprints, but also mistakes. We believe that results given by the conventional test are not reliable because correct answers are always shown to subjects during a test and this can cause a learning-while-testing side-effect, which can tamper results.

Thus, our experiment has two goals. The first one is to discover whether the body of an agent really make sense for humans. The second is to find whether the learning-while-testing effect really exists.

As a result of this experiment, we expect to see at least a weak preference for one of the methods of presenting information from most of subjects. No preference will mean that our test is not well designed and it can not “catch” the difference in two presentation styles. We also expect to see a learning-while-testing side effect, which is caused by the procedure of the conventional test and which should make results of the test less reliable, because we obviously should eliminate any learning during the test. Direction of future work will be discussed in Section 5 according to the results of the experiment.

3 Presenter Agent Experiment

The objective of the experiment is to investigate by using conventional IAT, whether the body of the ECA has an effect on subjects' attitudes towards two different methods of information mediation: full-featured ECA-based presentation and vocal presentation.

The experiment consists of two stages. In the first stage subjects were asked to learn two different stories from the presenter agent. The key difference between the two stories is the presence of the ECA on the screen. One story was presented by the female agent with a synthesized female voice and a rich set of non-verbal cues, while the other story was presented by the same female voice only. Both stories were accompanied with the same number of illustrations which were used later as categories and items in the IAT. Both stories are approximately the same size (200 and 226 words) and difficulty of memorization. The order of stories and method of presenting (which of two stories is presented by ECA and which by voice only) were chosen individually for each subject. Before the experiment subjects were told that the test will evaluate their attitudes towards interactions with agents. Subjects were not told that they should memorize information with will be presented. A screen-shot of the full-featured ECA-based presentation is shown in Fig. 1. As opposed to the previous method, in vocal presentation the agent does not appear on the screen, but the entire environment remains absolutely the same.

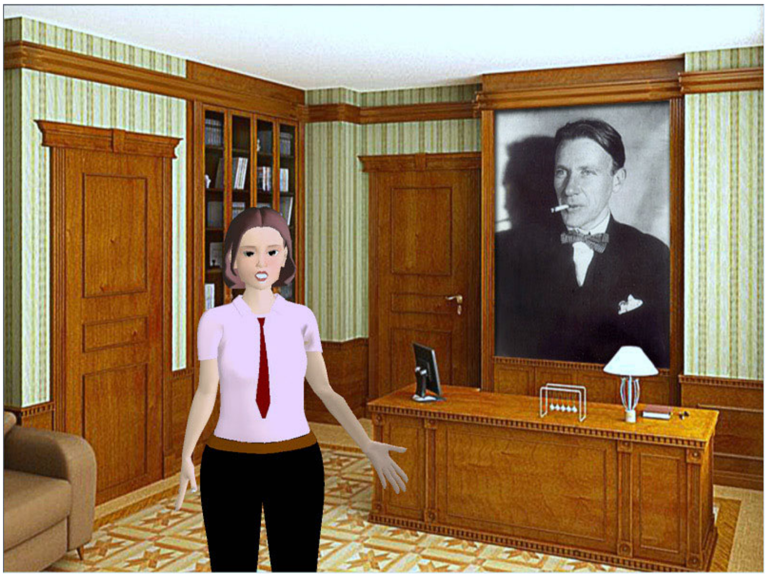


Fig. 1. Screen-shot of the full-featured ECA-based presentation

According to [1] people tend to unconsciously interpret the same voice as the same social actor. By choosing the same female voice for both stories we eliminated the necessity of comparing distinct social actors. Instead, two methods of information mediating from the same social actor were assessed by the IAT.

In the second stage, subjects' attitudes towards information mediation methods were assessed by the IAT. Thus, two key differences between our experiment and the conventional IAT are (as we previously mentioned in Section 2):

a) attempt to utilize indirect method of measurement with IAT (we assess methods of information mediation by assessing information blocks which were presented to subjects);

b) attempt to use the IAT on items which subjects are not familiar with (information blocks had been learned at most half-an-hour before the IAT).

3.1 Brief Description of Software

The Generic Embodied Conversational Agent (GECA) Framework [8] had been used as a platform for building the presenter agent. It provides some advantages compared to previously developed architectures. Components developed with different programming languages and even running on different operating systems can be easily integrated into the framework. Since the components can be distributed over a network those which require heavy computation can be sourced out to different computers and thus improve the overall system performance. The single-layer component hierarchy shortens the paths of decision making and eases the support of an agent's reactive behaviors. Synchronization of the different components is achieved by providing explicit temporal information and synchronization specifiers. Loose coupling of components allows for on-line switching and upgrading components easily.

In the experiment, ECA-based presentations were not interactive. GECA Scenario Markup Language (GSML) [8] was used to describe agent's behavior and environment. During the presentation subjects had no control over the agent, however the framework and GSML allows building of more complex dialog-style scenarios. GSML description of the agent behavior was played on the screen by the specialized 3D-player based on the Visage|SDK [9] platform.

The IAT was conducted using the Inquisit 3 [10] software. The Inquisit 3 is a general purpose psychological measurement software which allows us to administer almost any experiment where a high accuracy of stimuli presentation and response collection is required. The Inquisit can be used for web-based experiments as well as for laboratory tests. Data can be recorded to text files which can be easily imported by Excel or SPSS. De Clercq et al. [11] conducted an experiment which confirms the claimed millisecond accuracy of stimuli presentation and data accumulation by Inquisit.

3.2 Results

In total 10 subjects have participated in the experiment. Eight of them are students and two are administrative staff of Kyoto University, eight male and two female, all Asians, and all can listen and read in English fluently. Their results are shown in Figure 2.

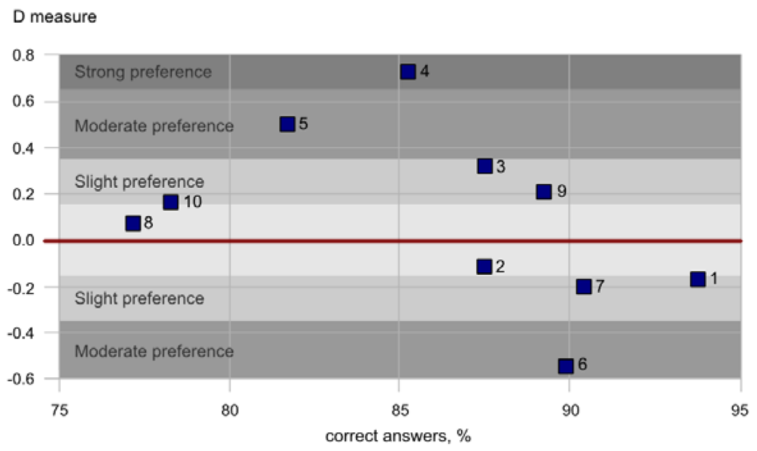


Fig. 2. Results of the IAT. X axis: number of correct answers in %; Y axis: result of the test (D measure)

The horizontal axis of this graph represents the number of correct answers and the vertical axis represents the D measure. The positive value of the D measure shows subject's preference towards full-featured ECA-based style of presentation and vice versa, the negative value shows preference towards voice-only presentation. All subjects had an error rate of less than 25% during the test. It should be noted that in this figure we do not distinguish between mistakes and misprints as well as between misprints in testing and reference categories. This means that the real number of meaningful mistakes might be less than shown on the graph.

The boundary values of subjects' preferences are as follows:

more than 0.65 – strong preference for the ECA-based presentation;
0.35 … 0.65 – moderate preference for the ECA-based presentation;
0.15 … 0.35 – slight preference for the ECA-based presentation;
-0.15 … 0.15 – no preference;
-0.35 … -0.15 – slight preference for the voice-only presentation;
-0.65 … -0.35 – moderate preference for the voice-only presentation.

Thus, subjects #8, and #2 do not show any significant preference for any kind of presentation. Subjects #3, #9, and #10 show slight preference for ECA-based presentations, however subjects #1, and #7 show slight preference for voice-only presentations. Subject #5, and #6 have moderate preference for ECA-based and voice-only presentations respectively. Finally, subject #4 shows a strong preference for the ECA-based presentation. Altogether, five subjects show significant preference for ECA-based presentations, three subjects show preference for voice-only presentations and only two subjects do not show any significant preferences. This conforms to the first part of our hypothesis.

One more fact which should be noted is related to a number of mistakes. As we can see from the graph, all subjects who show at least slight preference for the voice-only presentation (subject #1, #6, and #7), made less that 10% of mistakes and misprints. This is much better than the result of subjects who were attracted by the body of the agent.

An important fact is that the three subjects who show significant preference for one of presentation styles, reported that they could memorize correct answers during the test. Thus, they confirmed that they experienced the learning-while-testing effect. Some other subjects also experienced the same effect, however they did not report clearly about it. According to Fig. 3 subjects #2, #5, #6, #9, and #10 gave more correct answers in blocks 6 and 7 than in blocks 3 and 4. Please note, that for Fig. 3 and Fig. 4 we calculated only meaningful mistakes and misprints, eliminating misprints in reference categories. The total number of answers is 32. And as we can see, none of the subjects made zero mistakes. The best result was given by subject #6 in the blocks 6 and 7 – 29 correct answers.

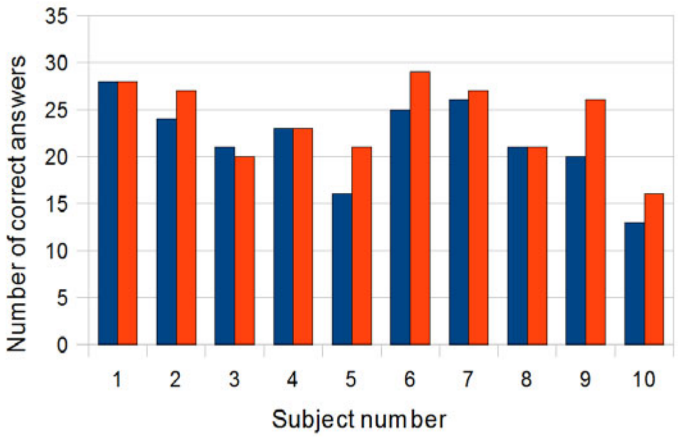


Fig. 3. Number of mistakes made in compared categories. Blue column – number of correct answers in block 3 + block 4; red column – block 6 + block 7.

Within ten minutes after the experiment subjects were asked to take the test once again. D-measures of the second test were not used and do not appear in Fig. 2, however we tried to find and analyze changes in the numbers of mistakes. These results are shown in Fig. 4.

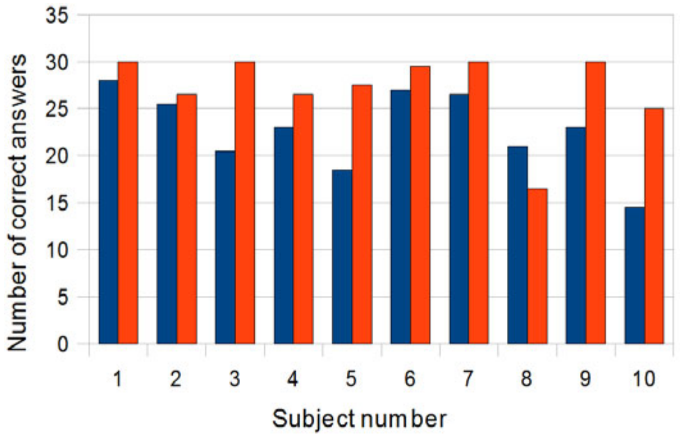


Fig. 4. Number of mistakes made in compared categories. Blue column – number of correct answers in the first test (mean of all pairing blocks); red column – second test (mean of all pairing blocks).

As we can see from the graph, subject #8 made quite a lot of mistakes in the second test, but all other subjects gave significantly more correct answers. In our opinion this means that subject learned correct answers during the first test and this confirms the second part of hypothesis.

4 Conclusions and Future Work

Taking into account the difference between conventional usage of an IAT and using an IAT for evaluating ECAs, several key issues can be defined and addressed in future work.

The one of conceptual issues is related to the fact that a conventional IAT deals with well known concepts while in the case of evaluating ECAs users deal with just-learned information, and this can cause mistakes in addition to misprints which are normal for the conventional IAT. In the conducted experiment subjects had only one chance to memorize information. Before the experiment they were not told that they should memorize information presented during the experiment, so they were expected to make mistakes.

On the other hand, during the conventional IAT wrong answers are always shown. Bearing in mind that in conventional IAT mistakes are not supposed to happen (misprints only, because subject deal with very familiar concepts only) this approach is very reasonable. However, for unfamiliar concepts, which we use in the experiment, it may cause a learning-while-testing side effect since each item is shown several times during the experiment. Thus, the first issue which should be addressed in future work is to modify the procedure of the test in order to minimize this effect.

One of the possible solutions which will be implemented in future experiment includes several steps. The first is to not show wrong answers during experiment. Essentially, this will minimize the learning-while-testing effect, but at the same time can distort final results. We propose to use more items for comparison concepts that are needed for measurements and eliminate before final calculations those of them which we answered with a high number of mistakes. This will help to administer final calculations without any further modifications.

Another issue which should be concerned in future work is the possibility of introducing a hardware response box for laboratory experiments which will be able to measure physiological data during the experiment, such as the force of pressing keys by subjects, their heart rate, blood oxygenation level, etc. This information will be used for validation purposes. Introducing the external hardware response box will also allow to measure time much more precisely in contrast to a PC in which accuracy can be affected by uncontrollable factors (e.g. operation system's high priority tasks).

We are also going to conduct a series of experiments in Japanese and Russian in order to allow more native speakers to participate. This will exclude at least one uncontrollable variable from the experiment which is related to fluent reading and understanding.

5 Summary

The goal of this work is to evaluate the potential possibility of using the Implicit Association Test where subjects' awareness of comparison concepts is less than in the case of conventional IAT, and to figure out possible issues related to this specific application. This paper presents results of the initial experiment where we used the conventional IAT procedure and scoring algorithms without any modifications. The data collected during the experiment shows how significant the difference between conventional usage of IAT and the proposed method is and which key issues should be addressed in future work. We showed our initial experiment which confirms our hypothesis about the effect of agent presence on the screen during presentation and procedural drawbacks of the conventional IAT in our particular circumstances. We made an analysis of D-measures and numbers of mistakes for each participant and outlined the directions of future research.

Acknowledgments

We highly appreciate contributions of all experiment participants. Our sincere gratitude to Divesh Lala, who carefully read the manuscript.

References

1. Nass, C., Steuer, J.S., Tauber, E.: Computers are social actors. In: Proceeding of the Computer-Human Interaction (CHI) 1994 Conference, pp. 72–78 (1994)
2. Cassell, J., Sullivan, J., Prevost, S.: Embodied Conversational Agents. MIT Press, Cambridge (2000)
3. Greenwald, A.G., McGhee, D.E., Schwartz, J.K.L.: Measuring individual differences in implicit cognition: The Implicit Association Test. Journal of Personality and Social Psychology 74, 1464–1480 (1998)
4. Greenwald, A.G., Nosek, B.A., Banaji, M.R.: Understanding and Using the Implicit Association Test: I. An Improved Scoring Algorithm. Journal of Personality and Social Psychology 85, 197–216 (2003)
5. Nosek, B.A., Banaji, M.R.: The go/no–go association task. Social Cognition 19, 625–664 (2001)
6. Sriram, N., Greenwald, A.G.: The Brief Implicit Association Test. Experimental Psychology 56, 283–294 (2009)
7. Penke, L., Eichstaedt, J., Asendorpf, J.B.: Single-Attribute Implicit Association Tests (SA-IAT) for the Assessment of Unipolar Constructs. Experimental Psychology 53(4), 283–291 (2006)
8. Huang, H., Cerekovic, A., Pandzic, I., Nakano, Y., Nishida, T.: The Design of a Generic Framework for Integrating ECA Components. In: Proceedings of 7th International Conference of Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, pp. 128–135 (May 2008)
9. http://www.visagetechnologies.com
10. http://millisecond.com/
11. De Clercq, A., Crombez, G., Buysse, A., Roeyers, H.: A simple and sensitive method to measure timing accuracy. Behavior Research Methods, Instruments and Computers 35, 109–115 (2003)

Robustness of Gamma-Oscillation in Networks of Excitatory and Inhibitory Neurons with Conductance-Based Synapse

Haibo Shi, Zhijie Wang*, Jinli Xie, and Chongbin Guo

College of Information Science and Technology, Donghua University,
Shanghai 201620, China
wangzj@dhu.edu.cn

Abstract. The stability of synchronization in E/I networks (networks of excitatory and inhibitory neurons) has been explored in many studies. However, how the robust synchronization occurs in highly heterogeneous real neural circuits has not been studied sufficiently. This paper considers the robustness of the synchronization in E/I network, and found that both the frequency and the synchrony of the network are highly dependent on the parameter settings. A short but indispensable inhibition delay, sufficiently fast decay of inhibitory synapses, and strong connections are critical to turn the E/I networks into robust gamma frequency oscillators. The networks under these conditions can achieve a level of robustness against heterogeneity as high as 30%, which is comparable to the level of heterogeneity observed in real neural systems.

Keywords: correlation coefficient, E/I networks, conductance-based synapse, heterogeneity.

1 Introduction

Gamma frequency oscillations are thought to provide temporal modulation for information processing in brain [1, 2]. They have been presumed to have a key role in cognitive functions such as feature binding and memory formations. Besides, disruption of gamma oscillation could underlie some psychiatric disorders [3].

Networks of excitatory cells and inhibitory cells (E/I network) are thought to be possible structures to sustain some experimentally observed gamma rhythm. The stability of correlative spike in E/I network has been verified, yet the robustness of E/I network subject to a heterogeneous drive is not well-studied [4]. Since heterogeneity in the driving current translates into the variability in the intrinsic spiking frequency, a physiologically plausible variance in the driving current (up to 35%) will break down the coherent oscillation [5].

Many studies used current-based synapses to study the synchronization of E/I networks [4]. As the conductance-based synapses with shunting effect of GABA synaptic inhibition helps to improve the robustness in interneuron networks against

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

the heterogeneity in tonic excitatory drive to some extend [6, 7], we used the conductance-based synapses in this paper. With this synapse model, synaptic current is relative to the post-synaptic membrane potentials referred to the excitatory and inhibitory reversal potentials [8]. With the conductance-based kinetics, synaptic currents have introduced rectification to counterbalance the variance in the corresponding frequencies caused by the heterogeneous stimuli. As a result, neurons in this model receive complementarily diverse synaptic current, rather than share identical synaptic current. The simulation results show that, under this regime, gamma-band oscillations can be maintained when the network is exposed to a high-level heterogeneity. We find, however, several requirements are needed to realize the robustness of the oscillation.

2 Model and Method

One hundred single-compartment neurons were arranged in all-to-all connectivity. The network consists of both E-cells and I-cells (excitatory and inhibitory neurons respectively) with a proportion of excitatory neurons versus inhibitory ones 4:1. The membrane potential of the neurons satisfies leaky-integrate-and-fire dynamics:

$$C_m \frac{dv_m}{dt} = I_{leak}(t) + I_{syn}(t) + I_{inj}(t) \ , \tag{1}$$

where C_m is the membrane capacitance, $I_{leak}(t)$ is the current due to the leak of the membrane, $I_{syn}(t)$ is the current describing the response of synaptic inputs to the neuron, and $I_{inj}(t)$ is the injected current which is responsible for the excitatory drive of the neuron. Since the membrane resistance R_m is normalised to 1, C_m can be replaced by the passive membrane time constant τ_m.

The leaky current is

$$I_{leak}(t) = -\frac{1}{R_m}[v_m(t) - V_{rest}] \ , \tag{2}$$

where V_{rest} is the resting potential. Whenever the potential reaches the threshold value V_θ, the neuron fires, and the potential resets to the rest potential V_{reset}.

Other than current-based synapse model, we hereby adapt the conductance-based synapse model, the synaptic current is determined by:

$$I_{syn,i}(t) = C_m[V_E - v_i(t)]\sum_{k=1}^{N_E} s_{ik} g_{E,k}(t) + C_m[V_I - v_i(t)]\sum_{k=1}^{N_I} s_{ik} g_{I,k}(t) \ , \tag{3}$$

where the potentials V_E and V_I (constants) are the reversal potentials ($V_{reset} < V_I < V_\theta < V_E$). The reversal potentials arise from specific neurotransmitters that cause the change to the membrane potential of the post-synaptic neurons. Their numerical values are identical to the membrane potentials of post-synaptic neurons at which neurotransmitters lead to no net ion fluxes. Whenever the pre-synaptic neuron fires, the conductance g (either $g_{E,k}$ or $g_{I,k}$) is reset to $g_{\max}$. s_{ik} is the coupling weight between neuron k and i .We set $V_E = -15$ mV and $V_I = -55$mV [9]. At intervals between spikes, the dynamics of $g_i(t)$ is given by a first–order kinetic:

$$\frac{dg_i(t)}{dt} = -\frac{g_{\max}}{\tau_{syn}} g_i(t) \ . \tag{4}$$

Each $g_{\max}$ is scaled accordingly to normalize the integrated conductance over the time course of synaptic activity. Numerically, $g_{\max}$ is divided by the area under the curve of $g_i(t)$ thus is dimensionless.

The excitatory input current to the network neurons consists of following two components

$$I_{inj} = I_\mu + I_\sigma \ , \tag{5}$$

where I_μ is the base current, and I_σ a random variable with normal distribution. As a consequence, each neuron is excited by an applied current with mean I_μ and standard variance I_σ. This will introduce input heterogeneity into this model. For a population of neurons that are non-connected with each other, diverse excitatory drives would cause dispersion in the neuronal firing frequencies. This variability in the intrinsic spiking frequency would exert a desynchronizing effect to the network. The heterogeneity in driving currents is measured by the ratio of variance versus mean I_σ / I_μ .

The coherence of the oscillation is estimated by the mean of the pairwise correlation of all neurons in the network [10]. The pairwise correlation is defined by :

$$\kappa_{ij}(T) = \sum\nolimits_{\tau=-T/2}^{T/2} y_i(\tau) y_j(\tau) \Big/ \sqrt{\sum\nolimits_\tau y_i(\tau) \sum\nolimits_\tau y_j(\tau)} \ , \tag{6}$$

where $y(\tau)$ indicates the idealized spike traces, which consists of trains of square pulses. Each pulse has unit height and width $= 0.2T$, and is centered at the spike peak as in Fig. 1.

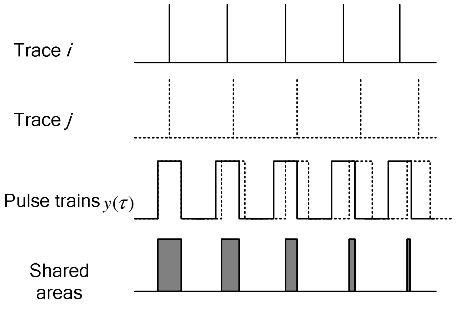


Fig. 1. The pairwise correlation demonstrated by the shared area of square pulses

3 Result

We started by making some exploration of the effect of parameter values on the robustness of the synchronization in E/I networks in parameter space. Parameters including the respective delay time of E-cells and I-cells and time constants of inhibitory and excitatory synapses are traversed to generate the corresponding coherence.

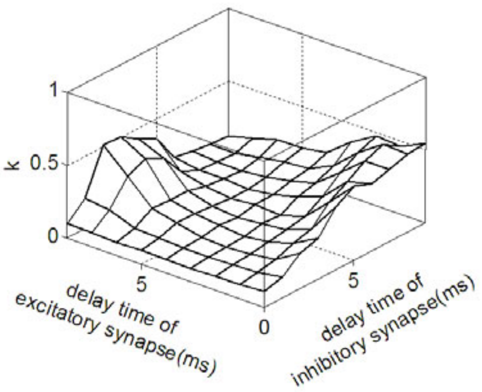


Fig. 2. Coherence v.s. the delay times of inhibitory and excitatory synapses

Relationship between the output correlation coefficient and delay times of excitatory and inhibitory synaptic currents is visualized in Fig. 2. The figure reveals that the latency time of inhibition plays a primary role in the synchronization of the network. Typically, an instantaneous inhibition will not produce a synchronized oscillation. The coherence soars when the onset of inhibition is delayed by 1-5ms (Fig. 3 A). A further increasing of inhibition latency will cause rhythmic bursting (Fig. 3 B) with the inter-bursting cycle in Gamma band and intra-bursting rate approximately 300Hz (Fig. 3 D). So only when the inhibition delay is not too long can the coherent oscillation be tuned to Gamma-band. Each box in the first row of Fig. 3 consists of two parts, a rastergram (upper) of spike train and a histogram (lower) which can imply the coherence via its steepness. The vertical axis is the neuron index, where neurons1-20 are inhibitory and neurons 21-100 are excitatory. The trajectory of membrane voltage of two representative neurons (an E-cell traced with solid line and an I-cell traced with dotted line) is sketched in Fig. 3 C.

Similarly, we examined the coherence as the function of the time constants of inhibitory synapses and excitatory synapses. The landscape of correlation coefficient is shown in Fig. 4 A. The decay time of inhibition has a strong effect on both the frequency and the synchrony of the resulting oscillation. When endowed with a slow decay, inhibition has a long duration but a small magnitude of ISPC. Under this situation, IPSP is too weak to suppress the inter-spike firing caused by the instantaneous excitatory pulse hence leads to either desynchronizing or ultra-fast spiking. The combination of large decay time constants of inhibition and excitation

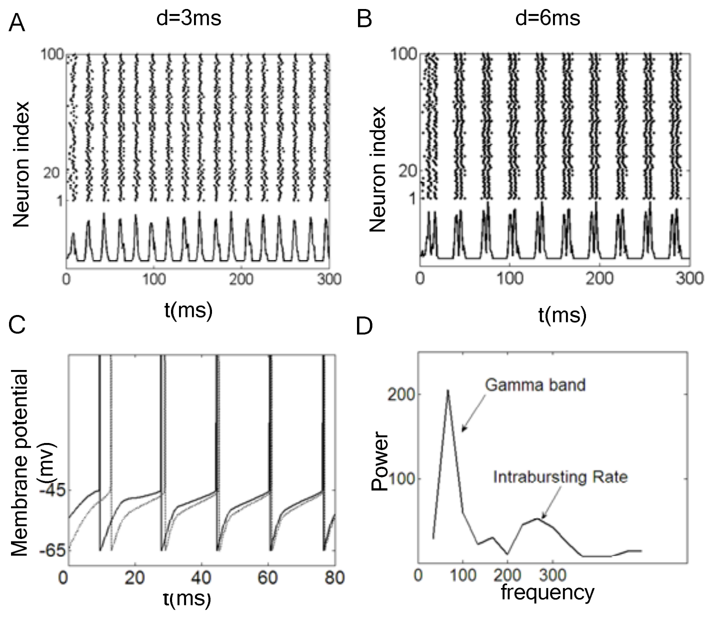


Fig. 3. Different oscillatory patterns with various delay times. Rastergram and histogram of spiking train with delay of (A) 3 ms and delay of (B) 6 ms. (C) Trajectory plot of membrane potentials of representative neurons in *A*. (D) Spectrum distribution of Rhythmic bursting in *B*.

gives rise to rhythmic bursting. The most undesired pattern occurs when the inhibition is slow and excitation is fast, in which neurons fire at ultrafast rates without coherence. Instead, synchrony is stubborn when decay time constant of inhibition is small, independent of the decaying rate of excitation. This optimized condition is consistent with the physiological observation, where the fast GABA-receptor mediated inhibitory post-synaptic current (IPSC) has a decay time constant of approximately 2 ms [11].

The behavior of the network with conductance-based synapse against the strength of connection and heterogeneity is illustrated in Fig. 4 B. In this set of simulation, the network is fed with excitatory tonic drive with different levels of heterogeneity (measured by the coefficient of variance $I_\sigma / I\mu$). The connection weight is also varied by the assigning the peak synaptic conductance with various values. Without a relatively strong connection, the correlation rate drops sharply along the axis of heterogeneity. In contrast, there is a ridge in the landscape of correlation coefficient corresponding to the strong-connection area. This implies that as the connection strength increases, the network develops a robust increasing tolerance against the heterogeneous input.

Finally, we systematically explored the relationship between the heterogeneity coefficients ($I_\sigma / I\mu$) and the coherence (Fig. 4 C). For each heterogeneity coefficient, we carry out the simulation of the network with all parameter values optimized according to the discussion in foregoing paragraphs. The mean coherence

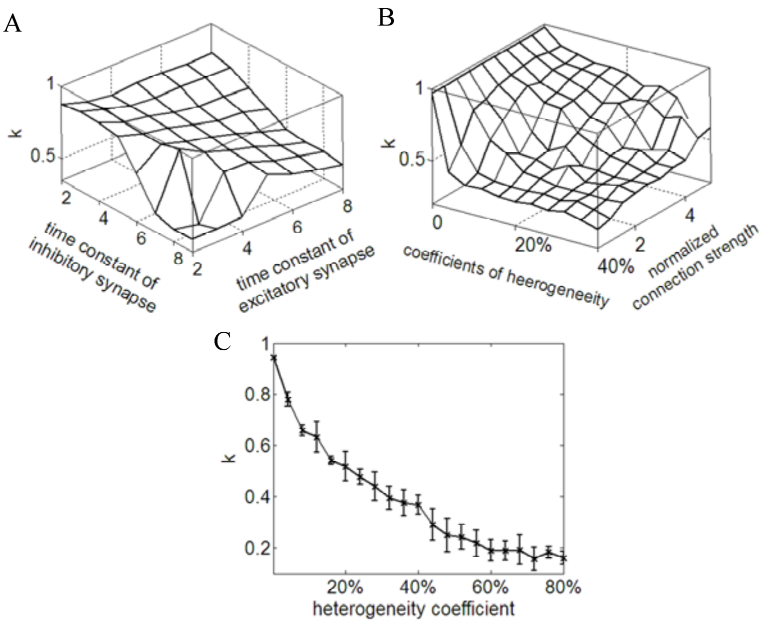


Fig. 4. The dependence of coherence on parameter values. (A) Coherence against time constants of excitatory and inhibitory synapses. (B) Correlation map in heterogeneity, connection strength-space. (C) Relationship between coherence and heterogeneity coefficient with optimized parameter values.

is estimated from twenty independent trials. The vertical bars centered at the mean values indicate corresponding standard deviations. When all requirements discussed above are met, coherent oscillations ($k > 0.4$) can be achieved for heterogeneity level around 30%. This indicates the level of robustness in this network is comparable to that observed in real neural systems.

4 Conclusion and Discussion

We examined the robustness of Gamma-oscillation in networks of excitatory and inhibitory neurons with conductance-based synapse. Since the conductance-based model is more physiologically realistic, the synchronizing effect it exerts in the simulation result may be of more biological interest. We also identified several conditions essential for the robust synchronizing. (1) A proper delay time of inhibition (1~5ms) is critical to maintain the correlation and to tune the oscillation frequency within Gamma band. (2) The decay time constant of inhibition is sufficiently short. Especially, a slow decaying inhibition is not compatible with fast excitation (with time constant approximately 2ms) for a high coherence. (3) Strengthening the overall connections can enhance the robustness when the network is subject to highly heterogeneous stimuli.

Acknowledgements

We acknowledge financial support from the National Natural Science Foundation of China (No. 61075105) and the China Ministry of Education (No.200802550007).

References

1. Gray, C.M., Singer, W.: Stimulus-specific neuronal oscillations in orientation columns of cat visual cortex. Proc. Natl Acad. Sci. 86, 1698–1702 (1989)
2. Buzsáki, G., Chrobak, J.J.: Temporal structure in spatially organized neuronal ensembles: a role for interneuronal networks. Curr. Opin. Neurobiol. 5, 504–510 (1995)
3. Spencer, K.M., et al.: Abnormal neural synchrony in schizophrenia. J. Neurosci. 23, 7407–7411 (2003)
4. Börgers, C., Kopell, N.: Synchronization in networks of excitatory and inhibitory Neurons with sparse, random connectivity. Neural Comput. 15, 509–538 (2003)
5. van Hooft, J.A., Giuffrida, R., Blatow, M., Monyer, H.: Differential expression of group I metabotropic glutamate receptors in functionally distinct hippocampal interneurons. J. Neurosci. 20, 3544–3551 (2000)
6. Buzsáki, G., Leung, L.S., Vanderwolf, C.H.: Cellular bases of hippocampal EEG in the behaving rat. Brain Res. Rev. 6, 139–171 (1983)
7. Vida, I., Bartos, M., Jonas, P.: Shunting inhibition improves robustness of gamma oscillations in hippocampal interneuron networks by homogenizing firing rates. Neuron 49, 107–117 (2006)
8. Burkitt, A.N.: A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input. Biol. Cybern. 95, 1–19 (2006)
9. Crunelli, V., Forda, S., Kelly, J.S.: The reversal potential of excitatory amino acid action on granule cells of the rat dentate gyrus. J. Physiol. 351, 327–342 (1984)
10. White, J.A., et al.: Synchronization and oscillatory dynamics in heterogeneous, mutually inhibited neurons. J. Comput. Neurosci. 5, 5–16 (1997)
11. Gerstner, W., van Hemmen, J.L., Cowan, J.D.: Synaptic mechanisms of synchronized gamma oscillations in inhibitory interneuron networks. Nat. Rev. Neurosci. 8, 45–56 (2007)

Probe the Potts States in the Minicolumn Dynamics[*]

Sanming Song and Hongxun Yao

School of Computer Science and Technology, Harbin Institute of Technology, 150001, Harbin, China
{smsong,yhx}@vilab.hit.edu.cn

Abstract. Minicolumn has been widely accepted not only as the basic structural element of the cortex anatomically, but also as the fundamental functional unit physiologically. And, it is believed by many theorists that the minicolumn may function as a Potts spine, only takes on finite discrete states. But its feasibility is unclear. In order to provide a biophysical evidence for the Potts assumption, a model is proposed to analyze the dynamics of minicolumn. With simulation, we found that Potts states may originate from the temporary high synchronization of neuronal subsets. Furthermore, after analyzing the average number of synchronous spiking neurons, we propose a novel and important assumption that, intrinsically-busting neurons may play a critical role in stabilizing the Potts states.

Keywords: minicolumn, Potts states, intrinsically-busting neuron.

1 Introduction

When analyzing the structure and function of the neocortex, a more macroscopic scale may be a better choice though neurons are the fundamental units. Not only for the size of neurons, 10^{11}, is computationally exhaustive, but also for the detailed synaptic and dendritic structures may be unimportant for the systematic analysis. The minicolumn assumption, first proposed by Mountcastle in 1957[1], and the later findings of anatomical and physiological evidences bridge the huge gap between the tiny neurons and the macro cortex.

But what is the functional role plays by the minicolumn? Or it is just a mediate motif towards the structure of the cortex. Neural theorists, like Treves[2] and Lansner[3], have assumed that they may function as Potts spins in statistical physics. That is, each minicolumn can only take on several finite discrete states, and it can be in only a single state at any time. But the feasibility and availability of the assumption is unclear. In order to understand the dynamics of minicolumn and provide a biophysical evidence for Potts assumption, a network model, which takes into consideration of the spiking characteristics of regular-spiking (RS) neurons, intrinsically-bursting (IB) neurons and fast-spiking (FS) inhibitory neurons, is constructed to explore the mechanisms underlying Potts assumption. Not only do we figure out the origination of Potts states, but also, we propose that IB neurons may be important in stabilizing the so-assumed "Potts states".

[*] This work is supported by National Natural Science Foundation of China (61071180).

© Springer-Verlag Berlin Heidelberg 2011

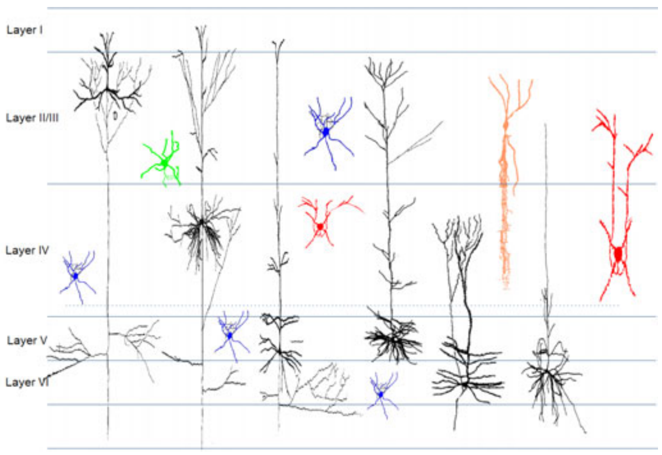


Fig. 1. Schematized graph of several typical neurons in the neocortex. Neurons with long dendrites and axons are pyramidal cell, and showed in black. Blue neurons are basket cells, and red for spiny stellate neurons. Orange for double bouquet cell and green for chandelier cell. Some neurons are redrawn from [7].

2 Minicolumn Model

The neurons in neocortex could be classified into spiny neurons and nonspiny nonpyramidal neurons, and the spiny neurons can be subdivided into pyramidal neurons and spiny stellate neurons. Almost all kinds of neurons are widespread in each layer, though with structural difference in synapses, dendrites, and the size of somas, except that spiny stellate cells mainly appears in layer IV. Though nonspiny nonpyramidal inhibitory neurons take up 15-30% of the total neuron populations, their subtypes are very complex and their roles are essential. According to their morphologies and spiking dynamics [4-5], we simply group them into three types, basket cells, bitufted cells, bipolar cells and chandelier cells, also see Fig.1. What's more, when the systematic dynamics is considered, we omit chandelier cells and use a generalized integrate-and-fire differential equation proposed by Izhikevich [6] to model the spiking dynamics of all the other inhibitory neurons.

According to the anatomical evidences [8] and computation speculations [9-10], we believe that 80 is typical for the total number of neurons in a single minicolumn. Also, the ratio between non-GABAergic excitatory cells and the GABAergic inhibitory cells is approximately 4:1. What's more, McCormick have reported that 35% of neocortical pyramidal cells are of the IB type[11], and the rest are RS neurons. So in the end, the neuronal distribution can be calculated, as showed in table 1.

In order to faithfully reflect the synaptic connections between neurons in layers, we adopted the results recorded from cat and rat by Thomson [12] (also see in Häusler et al. [13]). So now, we can construct a comprehensive network model for a minicolumn, as shown in Fig.2. See the caption for details.

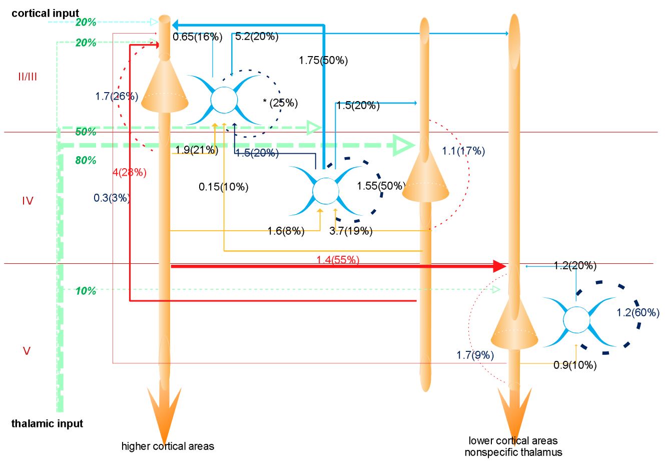


Fig. 2. Synaptic connections and weights in neocortical area, and data are adapted from[12] and [14]. Yellow cells stand for excitatory pyramidal neurons and blue for inhibitory interneurons. Note that, for convenience, the excitatory spiny stellate neurons in layer IV are incorporated into pyramidal cells. Arrows stand for synapses, blue for interneurons to pyramidal cells and yellow the reverse, red and deep blue for connections between pyramidal neurons and interneurons respectively. Intra-layer connections between the same kind of neurons are showed in dashed arcs, red for inter-pyramidal connections and deep blue for inter-interneuron connections. Also note the afferents and efferents. Afferents: cortical input sent into layer II/III and thalamic input mainly directed to layer IV. Efferents: the axons of pyramidal neurons in layer II/III directed into the superficial layers of higher cortical areas and the axons of deeper layers directed to lower cortical areas and nonspecific thalamus. * stands for Hopfield synapses.

2.1 Layer VI

It should be noted that, due to the following reasons, we simply omit layer VI in the network model.

Firstly, as it can be seen from Fig.1, pyramidal cells in layer VI collect information from layer IV and send their information to other tissues via their axon tufts. As Richardson et al [14] showed that, after external information transmitted to layer IV, forward information pathway passes layer II/III and targets layer V. Information in layer V either send to other areas directly, or send to thalamus and claustrum by relay of layer VI. So their main functions have been embodied in layer V.

Secondly, according to Callaway et al.[15], the synapses from thalamus to layer IV, from layer IV to II/III and then from layer II/III to layer IV of next higher cortex areas form the forward pathways. But the projection from pyramidal cells in layer V and layer VI to superficial layers have widespread synapse braches, in charge of the modal connection of neocortex, also see Fig.3. So the modality function is also embodied in layer V.

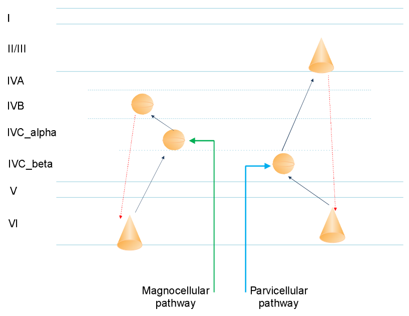


Fig. 3. Layer VI provides local modulatory feedback to superficial layers. The pyramidal cells that project to layer IVC_alpha can only receive inputs from layer IVB. Similarly, The cells that project to layer IVC_beta can only receive inputs fro layer II/III.

Table 1. Neuron distribution in a typical minicolumn

	Non-GABAergic		GABAergic	Layer_Total
	RS	IB	(FS)	
II/III	12	6	5	23
IV	20	10	7	37
V	10	6	4	20
Class_Total	42	22	16	80

Finally, Häusler et al. [13] have ever conducted simulation to compare the basic function of data provided by Binzeger et al.[16] and Thomson et al.[12], and their results showed that there is no radical differences between the two models, though the data provided by [12] lacks layer VI.

2.2 Afferent and Efferent

Sensory information from outside world primarily enter layer IV through the thalamus. Even though, the dominant afferents of the cortex areas are from the recurrent connections themselves. Besides afferents from the thalamus and other areas, neocortex also receives neuromodulatory inputs from brainstem, like norepinephrine, serotonin, acetylcholine and dopamine [14]. And for convenience, they are all considered to be weak external excitatory inputs in later simulations.

2.3 Dynamical Model

There are numerous neuron models in neuroscience that can be used to analyse the spiking characteristic of neurons and the system dynamics of networks. They are systematically classified into two types, with the first the biophysical model like Hodgkin-Huxley model and the second the integrate-and-fire model like Izhikevich model [6]. Here, the Izhikevich model is adopted for its biological plausibility and computational efficiency.

Assume $V_{l,\Xi}^{j}$ denotes the membrane potential of the j th neuron in class Ξ of layer l, where subscript Ξ represents the excitatory type, excitatory (e) or inhibitory (i), the ordinary differential equation of membrane potential can be written as

$$\begin{cases} \dfrac{dV_{l,\Xi}^{j}}{dt} = -0.04\left(V_{l,\Xi}^{j}\right)^{2} + 5V_{l,\Xi}^{j} + 140 - U_{l,\Xi}^{j} + I_{l,c} + \sum\limits_{k} w_{l,m}^{\Xi_j,\Xi_k} I_{syn}^{k} \\ \dfrac{dU_{l,\Xi}^{j}}{dt} = a\left(bV_{l,\Xi}^{j} - U_{l,\Xi}^{j}\right) \end{cases},$$

with the auxiliary after-spike resting

$$if \quad V_{l,\Xi}^{j} \geq 30 \quad then \quad \begin{cases} V_{l,\Xi}^{j} \leftarrow c \\ U_{l,\Xi}^{j} \leftarrow U_{l,\Xi}^{j} + d \end{cases}$$

Note that $U_{l,\Xi}^{j}$, the membrane recovery variable, which provides a negative feedback to membrane potential $V_{l,\Xi}^{j}$, is used to account for the activation of K^{+} and the deactivation of Na^{+} ion channel[6]。 $I_{l,\Xi}$ denotes the magnitude of external constant current that controls the global excitability of excitatory and inhibitory (Ξ) networks of layer l. And $w_{l,m}^{\Xi_j,\Xi_k}$ represents the synaptic strength between neuron j in layer l and neuron k in layer m. Parameters a, b, c, d are used to control the spiking behavior of different type of neurons, also refer to Fig.4.

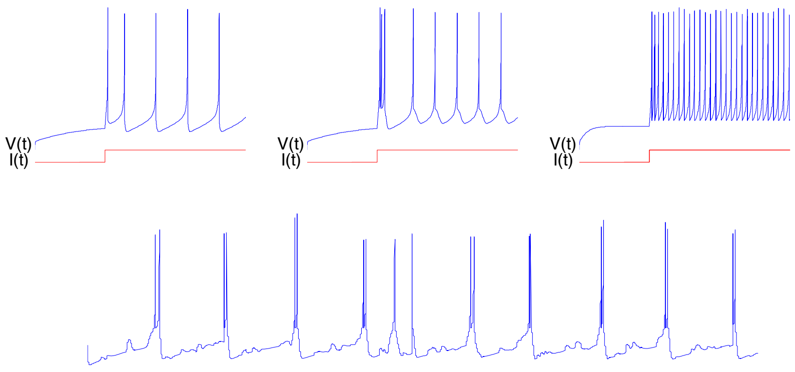


Fig. 4. Top: the spiking models of isolated RS, IB and FS neurons are shown from left to right respectively. Model parameters: RS (left): a=0.02, b=0.2, c=-65, d=8; IB (middle): a=0.02, b=0.2, c=-55, d=4 [6]; FS (right): a=0.1, b=0.2, c=-65, d=2. Bottom: typical firing style of an IB neuron in minicolumn.

3 Results and Discussions

The electrode-recorded data from Thomson et al.[12] only roughly reflects the connection weights between layers rather than detailed synaptic weights between neurons, also see Häusler et al.[13]. In order to introduce Potts states into minicolumn,

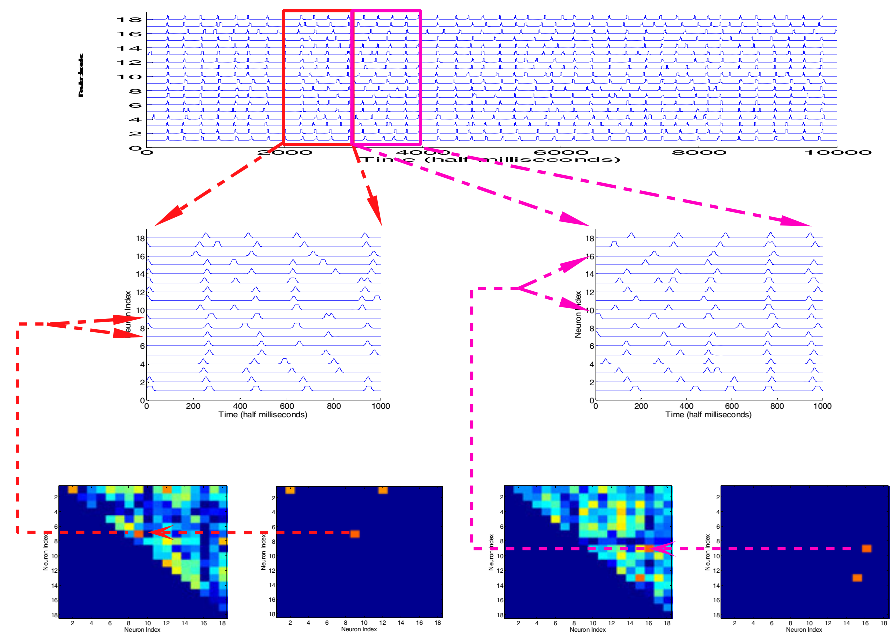


Fig. 5. The highly temorary synchronization represents the existence of Potts states. Top: the spiking of excitatory neurons in layer II/III. Middle shows the spiking activity in two time windows. Bottom shows the distance matrix between neuronal spiking series. See text for details.

more information should be stored in the layered minicolumn network. And we simply stored finite patterns into layer II/III according to Hopfield rule.

Treves et al.[17] has ever simply split the complex layered cortex into three layers, that is, a granular cell layer (layer IV) “has been inserted” into the original single pyramidal layer. It is said to be that the enlargement of memory capacity of layer II/III and layer IV lays the foundation for the emergence of human cognition. And also as discussed above, the recurrent connections from other areas mainly enter layer II/III. So we only store information in layer II/III.

EEG recordings show that only in Rapid Eye Movement or walking or complete action state, that the cortex exhibit rhythmic activity [18], when the thalamic inputs are inhibited. And what’s more, only when the systematic dynamics is rhythmic, can we analyze the Potts states efficiently. So we set the afferents of layer II/III far stronger than that of other layers. By the way, the weak neuromodulatory inputs from brainstem are incorporated into the weak afferents.

3.1 Potts States

To scrutinize the temporary spiking synchronization of neurons, we compartmentalize the spiking series into short time windows, as shown in Fig.5. Then we calculate the distance between each pair of neuronal spiking series. Now that the distance is symmetric, only the above triangle of the matrix is shown for convenience. As it can be easily seen from Fig.5, only a subset of neurons are spiking synchronously. In the first window, neuronal pairs (1, 2), (1, 12) and (7, 9) have very similar firing timing in that particular period, with the pair (7, 9) firing in best synchronization. Similarly, neuronal pairs (9, 16), (13, 15) are the most synchronous pairs in the second window.

And several interesting solutions can be drawn directly from these simulations. Firstly, it is the highly synchronization of a subset of neurons underlies Potts states. When all excitatory neurons in layer II/III are in-phase, there would be only two discrete states, on or off, for each minicolumn. If so, the memory capacity of human brain would be too restricted for the cognitive behaviors. What's worse, no latching phenomenon [2] would appear. But if all neurons are asynchronous, or phase-locked, the dynamics of minicolumn would be chaotic. If so, the semi-rhythmic activity appeared in several status would disappear.

Secondly, all neurons in minicolumn are almost equal-active. That is, their average firing activities are stable, which not only excludes the assumption of Potts states by the number of total spikes of all neurons in a short period, but also demonstrates firing stability of minicolumn.

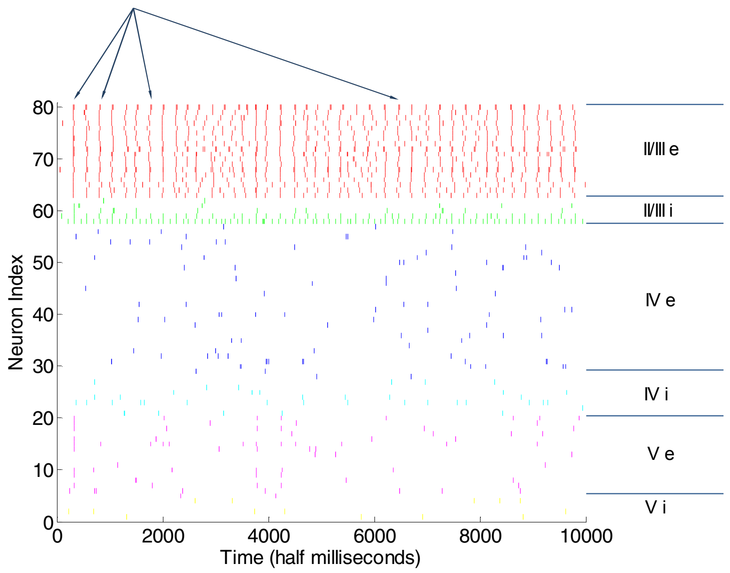
Lastly, we can simply estimate the Potts states each minicolumn has. As shown in Fig.7, the average number of most synchronous neurons is 3.8143, with standard deviation 1.8205. If the number of synchronous spiking neurons is simply taken to be 4, then there would be 3e+3 Potts states. If the surface area of human cortex area is 2400cm^2, [14] and the diameter of minicolumn is 0.05mm [19], there would be 1.2e+8 minicolumns in total. And according to [20], if the neurons are fully connected (only roughly estimate the upper bound), then the total patterns that could be stored in our neocortex would be 1.5e+14, almost infinite.

3.2 Intrinsically-Bursting Neurons

As discussed above, besides RS neurons, IB neurons also take up a large part of the excitatory neurons. And it is curiously that this obvious phenomenon sometimes even has been neglected, like Golomb et al.[21].

To probe the probable role for IB neurons, we also run the network when all the excitatory neurons are RS neurons. And a comparison between minicolumn networks with and without IB neurons is showed in Fig.6 and Fig.7. There is nearly no difference between the rhythmic spiking activity between two network models. But in Fig.7, we found that not only the average number of most synchronous spiking neuron, but also the standard deviations of the network with IB neurons are smaller. With the former, the maximum amplitude of rhythmic waves is lower, and with the later, the Potts states are more uniform and stable.

The more uniform synchronous group-spiking shows that the firing activity of excitatory neurons are affected by stronger synaptic inputs, and we think the bursting behavior of IB neurons (Fig.4, bottom) is an ideal choice. Also, when FS inhibitory neurons are excited by stronger presynaptic neurons, they depress the excitability of the whole network, lowering the amplitude. So it comes about a probable role that IB neuron plays in the neocortex, enhancing the spiking stability of minicolumns.



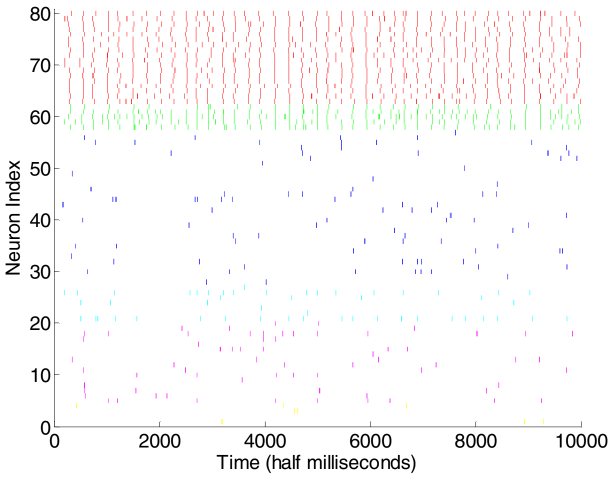


Fig. 6. Minicolumn dynamics with (top) and without (bottom) IB neurons. And there is nearly no difference between the rhythmic activity of two models. Note that the afferents from thalamus are inhibited, so neurons in layer IV seldom firing. e stantds for excitatory, I for inhibitory.

It should be noted that French et al. [22] have ever proposed that IB neurons are primarily in charge of the synchronous spiking of the total network. But as it can be seen from Fig.6, even when the IB neurons are excluded from minicolumn, the systematic rhythmic synchronization activity also appears.

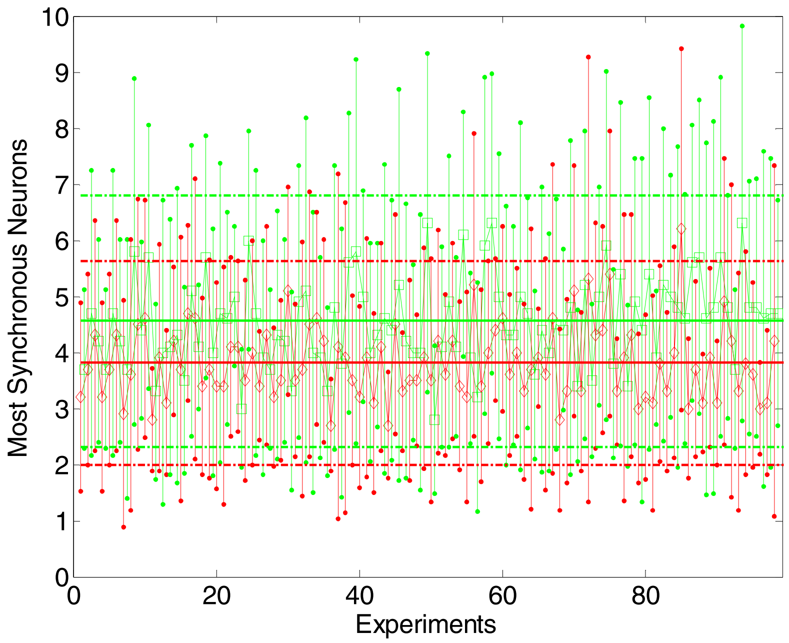


Fig. 7. Comparison of the number of the most synchronous spiking neurons in networks with (red) and without (green) IB neurons. The solid line represents the average number, and the dashed line shows the standard devidations. The statistics is 3.8143 ± 1.8205 when IB neurons are involved, and 4.5551 ± 2.2498 when not.

Perhaps IB neurons have other particular physiological or other biochemical functions, but the conclusion we draw here provides a novel viewpoint for their functional roles.

4 Conclusions

In this paper, we try to explore the feasibility and availability of so-assumed functional role that minicolumn plays in the systematic dynamics of neocortex, Potts states. We found that Potts states probably originate from the temporarily high spiking synchronization of a subset of neurons. Furthermore, we analyzed the role played by the IB neurons and proposed that it may have much to do with the stability of Potts states. Future work includes how the most synchronous spiking pairs evolves and how the minicolumn functions when thalamus inputs appears and when the proportion of IB neurons changes.

References

1. Mountcastle, V.B.: Modality and topographic properties of single neurons of cat's somatic sensory cortex. J. Neurophysiol. 20, 408–434 (1957)
2. Treves, A.: Frontal latching networks: A possible neural basis for infinite recursion. Cognitive Neuropsychology 22(3-4), 276–291 (2005)

3. Lansner, A.: Associative memory models: from the cell-assembly theory to biophysically detailed cortex simulations, Rev. Trends in Neuroscience 32(3), 178–186 (2009)
4. Gupta, A., Wang, Y., Markram, H.: Organizing principles for a diversity of GABAergic interneurons and synapses in the neocortex. Science 287, 273–278 (2000)
5. Markram, H., Toledo-Rodriguez, M., Wang, Y., Gupta, A., Siblberberg, G., Wu, C.: Interneurons of the neocortical inhibitory system. Nature Review Neuroscience (5), 793–807 (2004)
6. Izhikevich, E.M.: Which Model to Use for Cortical Spiking Neurons? IEEE Trans. Neural Networks 15, 1063–1070 (2004)
7. DeFelipe, J., Farinas, I.: The pyramidal neuron of the cerebral cortex: morphological and chemical characteristics of the synaptic inputs. Prog. Neurobiol. 39, 563–607 (1992)
8. Peters, A., Sethares, C.: Myelinated axons and the pyramidal cell modules in monkey primary visual cortex. J. Comp. Neurol. 365, 232–255 (1996)
9. Peters, A.: The morphology of minicolumns. In: Blatt, G.J. (ed.) The Neurochemical Basis of Austim, pp. 45–68. Springer Science & Business media, LLC (2010)
10. Buxhoeveden, D.P., Casanova, M.F.: The microcolumn hypothesis in neuroscience. Brain 125, 935–951 (2002)
11. McCormick, D.A., Connors, B.W., Lighthall, J.W., Prince, D.A.: Comparative electrophysiology of pyramidal and sparsely spiny stellateneurons of the neocortex. J. Neurophysiol. 54, 782–806 (1985)
12. Thomson, A.M., West, D.C., Wang, Y., Bannister, A.P.: Synaptic connections and small circuits involving excitatory and inhibitory neurons in layers 2–5 of adult rat and cat neocortex: triple intracellular recordings and biocytin labeling in vitro. Cerebral Cortex 12(9), 936–953 (2002)
13. Häusler, S., Maass, W.: Motif distribution, dynamical properties, and computational performance of two data-based cortical microcircuit templates. J. Physiology 103, 73–87 (2009)
14. Richardson, K.A., Fanselow, E.E., Connors, B.W.: Neocortical Anatomy and Physiology. In: Engel, J., Pedley, T.A. (eds.) Epilepsy: A Comprehensive Textbook, 2nd edn., pp. 323–336. Lippincott-Williams & Wilkins (2008)
15. Callaway, E.M.: Feedforward, feedback and inhibitory connections in primate visual cortex. Neural Networks 17(5-6), 625–632 (2004)
16. Binzegger, T., Douglas, R.J., Martin, K.A.: A quantitative map of the circuit of cat primary visual cortex. J. Neurosci. 24(39), 8441–8453 (2004)
17. Treves, A.: Computational constraints that have favoured the lamination of sensory cortex. J. Comput. Neurosci. 14, 271–282 (2003)
18. Jablonski, P., Poe, G.R., Zochowski, M.: Structural network heterogeneities and network dynamics: A possible dynamical mechanism for hippocampal memory reactivation. Phys. Rev. E 75(011912), 1–7 (2007)
19. Favorov, O., Kelly, D.G.: Minicolumnar organization within somatosensory cortical segregates:II. Emergent Functional Properties. Cerebral Cortex 4, 428–442 (1994)
20. Kanter, I.: Potts-glass models of neural networks. Phys. Rev. A 37, 2739–2742 (1988)
21. Golomb, D.: Slow excitation supports propagation of slow pulses in networks of excitatory and inhibitory populations. Phys. Rev. E 65(061911), 1–16 (2002)
22. French, D.A., Gruenstein, E.I.: An integrate-and-fire model for synchronized bursting in a network of cultured cortical neurons. J. Comput. Neurosci. 21, 227–241 (2006)

Dependence of Correlated Firing on Strength of Inhibitory Feedback

Jinli Xie*, Zhijie Wang, and Haibo Shi

College of Information Science and Technology, Donghua University,
Shanghai 201620, China
xjl802@mail.dhu.edu.cn

Abstract. Correlation coefficient is a way to capture the structure of correlation, which is widely used for analysis and interpretation of multiple simultaneously recorded spike trains. We study the correlation coefficient of a global delayed feedback network with a population of excitatory neurons and a population of inhibitory neurons and show the contribution of the inhibitory feedback to the oscillations and correlation. We find that, with the increase of the strength of the inhibitory feedback, the mean firing rate of the excitatory neurons decreases monotonically, while the correlation coefficient first goes down, and then goes up when the excitatory neurons oscillate and the spectrum shows significant resonance. The non-monotonic relationship between the correlation coefficient and the strength of inhibitory feedback is thus due to the combined effect of the mean firing rate and the power of oscillations.

Keywords: correlation, oscillation, inhibitory feedback.

1 Introduction

Many neurons often share common fluctuations in both field potentials and neural firing, which can be induced by both common external stimuli or shared internal connections [1-7]. Previous related studies have shown that temporal correlation is fundamental tools for encoding and exchanging information for neuronal information processing [8-10]. Therefore, it is essential that we gain a thorough understanding of correlation in the brain and its impact on population coding. Progress has been made on the temporal scales and spatial extent of correlation of spiking activities [8,9,11], however, the interpretation of the correlation is still difficult, especially in large networks [11].

Rhythmic spiking activities are often found in networks of excitatory neurons and inhibitory neurons [2,5,12], the excitatory neurons synchronize the inhibitory neurons and vice versa. The interplay between the two cells groups may lead to Gamma oscillations, which have been the subject of intense research effort in many studies [4,6,12]. Neurophysiological studies have suggested that synchronous oscillatory activities may be evoked by inhibition originating from inhibitory interneurons [3,4,13]. Therefore, in a network with excitatory and inhibitory neurons, a delayed inhibitory feedback pathway is critical for the stochastic oscillations.

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

In the present paper we consider a population of identical, uncoupled excitatory neurons with the correlated external input, receiving a common inhibitory synaptic input from a population of inhibitory neurons. This architecture is widely used in discussing the oscillatory, synchronized firings of cortical pyramidal neurons [2,5]. However, the effect of inhibitory feedback connections on the correlation is rarely discussed in theoretical or numerical studies [11].

In our former studies, the presence of non-monotonic relationship between correlation coefficient and the strength of feedback gain could be proved in a network which has only one inhibitory neuron in the second layer [14,15]. In the present paper, we find that the correlation coefficient of the excitatory neurons could still be non-monotonically changed by increasing the strength of feedback gain when we replace a single inhibitory neuron by a population of inhibitory neurons. The non-monotonic relationship reported here is quantitatively and qualitatively similar as the one in our former studies. Furthermore, we find that both the firing rate and the oscillations of populations of neurons are important and should be involved to understand the correlation of the excitatory neurons.

2 Network Model

We use leaky-integrate-and-fire (LIF) model for simulations and numerical analysis. The LIF model is an extremely useful description of neuronal activity, which describes the dynamics of the membrane potential by Eq.1 and a simple spike-and-reset rule: every time the potential reaches a firing threshold ν_T, the neuron fires and resets to the reset potential ν_R. After firing, the neuron is in an absolute refractory state for time τ_R.

$$\frac{dv(t)}{dt} = -v(t) + I(t) \ , \tag{1}$$

where *v(t)* and *I(t)* are the membrane potential and afferent current respectively. Here time is measured in units of the membrane time constant and the resistance of the cell membrane is normalized to one. The neuronal parameter values are set to: ν_T=1, ν_R=0, τ_R=1.

The output spike train of LIF model is:

$$y(t) = \sum_j \delta(t - t_j) \ , \tag{2}$$

where the t_j are the successive firing instants. Based on the LIF model, the structure of the network contains two layers of N_E=4N/5 excitatory and N_I=N/5 inhibitory neurons. We use these proportions because there are about four times more excitatory than inhibitory neurons in the cortex. All excitatory neurons receive correlated external input drive *Ii*. The inhibitory neurons provide a common negative input I_{IE} to all the excitatory neurons. The synaptic time constants have two different values t_E or t_I, depending on whether neuron is excitatory or inhibitory.

$$I_i(t) = \mu + \sigma\left[\sqrt{1-c}\,\xi_i(t) + \sqrt{c}\,\xi_c(t)\right] + I_{IE} \ , \tag{3}$$

$$I_{IE} = G\int_{\tau_D}^{\infty} \alpha(\tau) \sum_{j=1, j\in I}^{N/5} y(t-\tau)d\tau \ , \tag{4}$$

$$\alpha(t) = \frac{t-\tau_D}{\tau_I^2}\exp\left[-\frac{t-\tau_D}{\tau_I}\right] , \tag{5}$$

where μ denotes the base current. $\sigma[\sqrt{1-c}\ \xi_i(t)+\sqrt{c}\ \xi_c(t)]$ stands for the external noise with intensity σ, and consists of two noise processes: $\xi_i(t)$ are the individual noise components, and $\xi_c(t)$ the shared component. Input correlation coefficient c could determine the degree of correlated external input. I_{IE} represents the common negative input to the excitatory neurons. G is the total strength of the feedback pathway which defines the feedback input by multiplying the convolution of a delayed a function and the spike trains of all the inhibitory neurons. Here t_D is the transmission delay of the feedback loop. The inhibitory neurons are drived by all the excitatory neuron with a common input I_{EI}.

$$I_{EI} = \mu + \int_{0}^{\infty} \alpha(\tau) \sum_{j=1, j\in E}^{4N/5} y_j(t-\tau)d\tau \ , \tag{6}$$

$$\alpha(t) = \frac{t}{\tau_E^2}\exp\left[-\frac{t}{\tau_E}\right] . \tag{7}$$

We will mainly interest in the subthreshold regime. The neurons fire only with the presence of external input. The values of all other parameters are: τ_D=4ms, τ_E=2ms, τ_I=10ms, μ=0.9, σ=0.3, N=100, c=0.2. Eq. 1 is integrated by a simple Euler scheme with a time step of 10^{-5}sec.

3 Methods and Results

Since two coupled populations of excitatory and inhibitory neurons can reveal oscillatory activity, we first calculate the spike train power spectral density of the excitatory neurons to study the rhythmic spiking activities of the network. One excitatory neuron is selected to reveal the properties of the oscillations. The power spectrum of spike trains of neuron i is calculated by:

$$S(f) = \langle \tilde{y}_i \tilde{y}_i^* \rangle \ , \tag{8}$$

$$\tilde{y}_i(f) = \frac{1}{\sqrt{L}}\int_0^L e^{-i\omega t} y_i(t)dt \ , \tag{9}$$

where $\tilde{y}_i$ is the Fourier transform of the spike train, $\tilde{y}_i^*$ denotes the complex conjugate of $\tilde{y}_i$.

Fig. 1 shows the result of computer simulations for the spike train power spectral density of an excitatory neuron from the network when G=-0.5. With the presence of

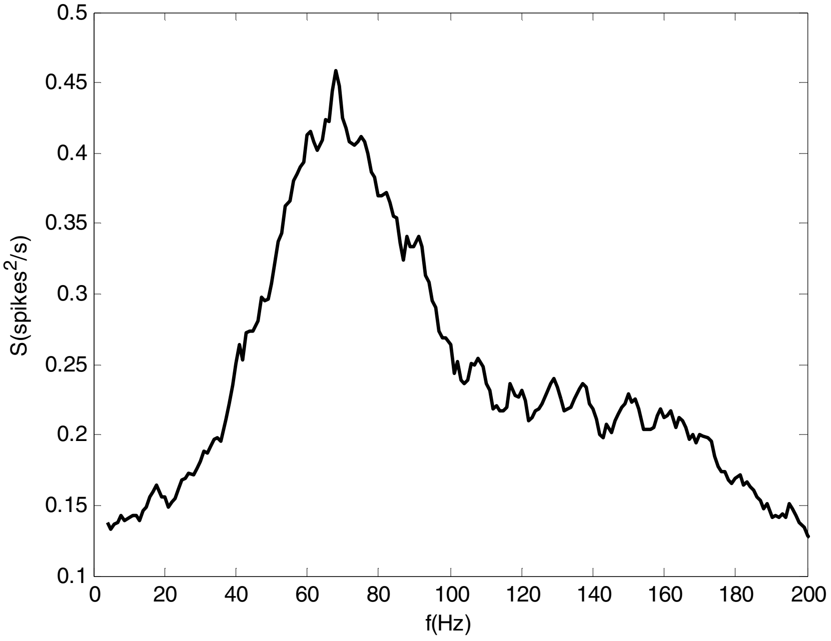


Fig. 1. Spike train power spectral density of an excitatory neuron

global delayed feedback and correlated external input to the excitatory neurons, we obtain a clear resonance of the spectrum in the gamma frequency band. This result is corresponding to the former studies [3-5]. In order to measure the intensity of the oscillations of the excitatory neurons and obtain a compared result, the amplitude of the resonance can be quantified by introducing the statistics:

$$\Gamma_{f_1,f_2} = \int_{f_1}^{f_2} S(f)df \ . \tag{10}$$

Over our data ensemble, we have $\Gamma_{f1,f2}$ for frequencies $\Gamma_{40,100}$, since the resonance peak of *S(f)* in fig. 1 is centralized in this band. Furthermore, we use $\Gamma_{40,100}$ to quantify the shift in power with inhibitory feedback gain *G*. As shown in fig. 2, in the subthreshold regime, we analysis the relationship between $\Gamma_{40,100}$ and *G*. Over the whole range of *G*, $\Gamma_{40,100}$ decreases monotonically with the increasing of *G*. When *G* is small, in other words, the inhibitory feedback is strong, the value of $\Gamma_{40,100}$ keeps high and stable, indicating continuing oscillations of the excitatory neurons with strong inhibitory feedback. When *G* is increased, the value of $\Gamma_{40,100}$ drops quickly, which reveals that the peak value of oscillations in the presence of weaker inhibitory feedback is decreased. Finally, the oscillations disappear gradually when *G* tends to zero, leading to relatively small $\Gamma_{40,100}$.

Power spectral density reveals the properties of the spike trains in frequency domain. To discuss the firing activities in time domain, we use pairwise spike train correlation

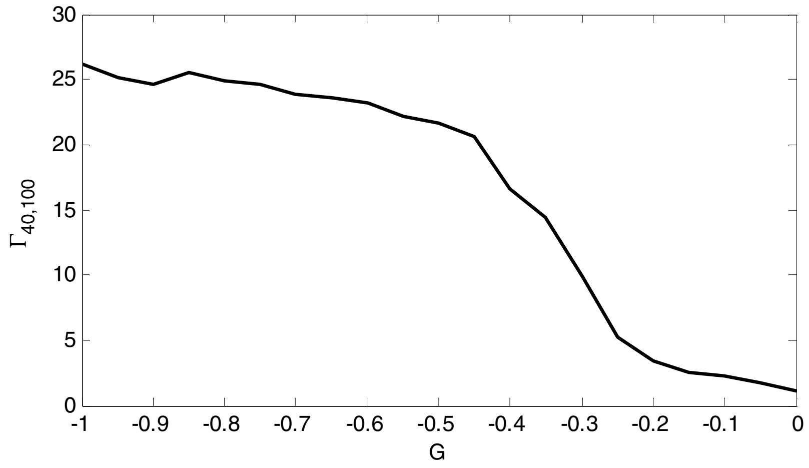


Fig. 2. Power $\Gamma_{f1,f2}$ near the resonance peak (40-100 Hz) as a function of *G*

in this paper to estimate the correlation coefficient. Pairwise spike train correlation is a widely used method to investigate the correlated firing of neurons. It is calculated by the ratio of the area of the cross-correlogram (CCG) and the geometric mean area of the auto-correlogram(ACG) with certain range of integral window T.

The expressions of CCG and ACG are: [9,10,11],

$$CCG_{ij}(\tau) = \frac{\sum_{k=1}^{M}\sum_{t=0}^{L} y_i^k(t)y_j^k(t+\tau)}{M(L-|\tau|)\sqrt{\lambda_i\lambda_j}} , \tag{11}$$

$$ACG_{jj}(\tau) = \frac{\sum_{k=1}^{M}\sum_{t=0}^{L} y_j^k(t)y_j^k(t+\tau)}{M(L-|\tau|)\sqrt{\lambda_j\lambda_j}} , \tag{12}$$

where M is the number of trials, L is the duration of every trial, λ_i and λ_j are the firing rates of neurons *i* and *j*. L-|τ| is used to correct for the degree of overlap of the two spike trains [9,10]. Once the geometric mean spike rate $\sqrt{\lambda_i\lambda_j}$ is divided CCG and ACG end up with units of coincidences per spike.

To weaken the influence of slow fluctuations in neuronal response, we correct all CCGs and ACGs by subtracting a shift predictor SPT:

$$SPT_{ij}(\tau) = \frac{\sum_{k=1}^{M}\sum_{t=0}^{L} y_i^k(t)y_j^{k'}(t+\tau)}{M(L-|\tau|)\sqrt{\lambda_i\lambda_j}} , \tag{13}$$

where k′=k+1(k<M) or k′=1(k=M).Therefore, we can obtain the expression of the pairwise spike train correlation C_{ij} based on eq. (11)-(13):

$$C_{ij}(T) = \frac{\sum_{\tau=-T}^{T} CCG_{ij}(\tau) - \sum_{\tau=-T}^{T} SPT_{ij}(\tau)}{\sqrt{[\sum_{\tau=-T}^{T} ACG_{ii}(\tau) - \sum_{\tau=-T}^{T} SPT_{ii}(\tau)][\sum_{\tau=-T}^{T} ACG_{jj}(\tau) - \sum_{\tau=-T}^{T} SPT_{jj}(\tau)]}} . \quad (14)$$

When T is large enough, the correlation coefficient of the network (*Cor*) can be calculated by averaging the pairwise spike train correlation over all the excitatory neurons. Here M=100, L=10s, T=100ms.

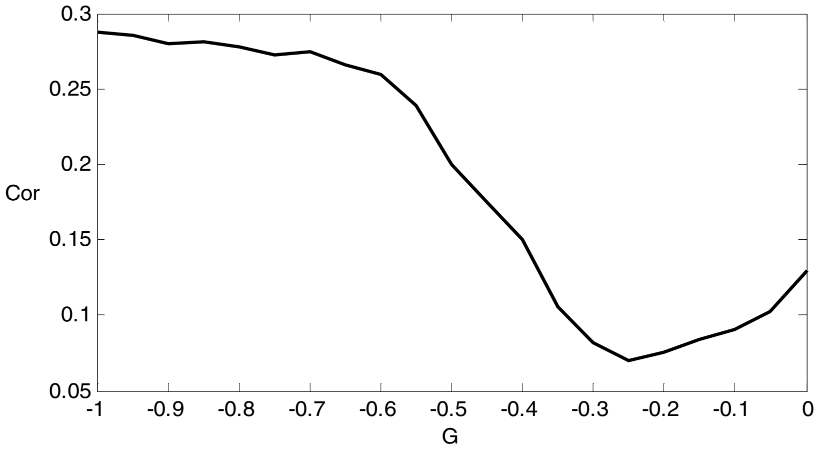


Fig. 3. Correlation coefficient *Cor* as a function of *G*

The numerical result for *Cor* versus *G* is presented in fig. 3. Non-monotonic correlation coefficient function is observed: for moderate values of *G* the curve shows a notch. This non-monotonic relationship between correlation coefficient and inhibitory feedback gain is demonstrated before when we use a single inhibitory neuron in the second layer [14,15]. Here a set of 20 LIF inhibitory neurons are introduced to investigate the interaction between populations of neurons. Since it is difficult to determine the nature of states in large neural networks, more attention should be given to the simulations in this field. Remarkably, the non-monotonic curve is preserved when we expand the scale of networks. Adding more inhibitory neurons does not affect the results, as shown in fig. 3.

In order to compare the results of the power of the oscillations and the correlation coefficient and identify the effects of inhibitory feedback, $\Gamma_{40,100}$ (gray pluses) and *Cor* (black line) as functions of *G* are normalized and replotted in fig. 4. We also incorporate the mean firing rate (*R*) of the excitatory neurons with varying values of *G* in fig. 4 (gray crosses). Both the *Cor* and the *R* functions reach smaller values with weaker inhibitory feedback gain's (larger *G*'s). The decreasing of firing rate leads to the drop of the correlation coefficient, corresponding to the conclusion of previous studies: the correlation is directly proportional to the firing rate [10]. In contrast, we also find *Cor* begins to increase when inhibitory feedback becomes stronger (smaller *G*'s); thus the upward trend of the curve of *Cor* does not reflect the effect of firing rate

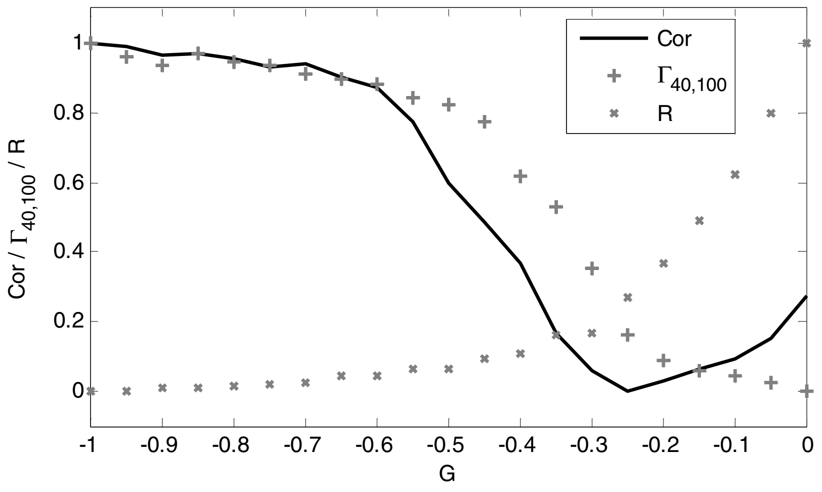


Fig. 4. Normalized *Cor*, $\Gamma_{40,100}$ and *R* as functions of *G*

on the correlation. This is because when *G* decreases, $\Gamma_{40,100}$ begins to rise, i.e. the excitatory neurons oscillate apparently. The increment in $\Gamma_{40,100}$ is much bigger than the decrement in *R*. The correlation coefficient thus increases owing to rhythmic spiking activities. Moreover, *Cor* drops slower than *R* dose with larger *G*'s. This is also due to the effect of mild oscillations, which are indicated by small values of $\Gamma_{40,100}$.

4 Conclusion and Discussion

The neural systems are often required to perform complex computations demanding strong correlations [8]. Experimental recording of neuronal activity often show large oscillatory response, indicating periodic and synchronized spiking occurs across distant neurons [1,6,12]. These synchronized oscillations might generate short time scale or long time scale correlations [11], which allow patterns of populations' activity to propagate more efficiently to downstream targets.

In the present paper, we have shown that the relationship between correlation coefficient of the network and the strength of the inhibitory feedback gain is non-monotonic. The decrease of the firing rate of the excitatory neurons by adding inhibitory feedback input lead to the decline of the correlation coefficient at weak inhibitory feedback gain. With the increase of the strength of the inhibitory feedback gain, the network begins to oscillate in gamma frequency; the periodic components of the firing activity of the excitatory neurons give rise to the correlation coefficient of the network, regardless of the effect of the firing rate. In conclusion, our results suggest that correlations are related to two parameters: the firing rate of the neurons capturing the features of the stimuli, which are proportional to the correlations for non-interactive neurons; and the inhibitory feedback gain, which describes the strength of the interior connections of the network, could enhance the correlations by inducing periodic oscillatory activity of neurons.

Acknowledgments. We acknowledge financial support from the National Natural Science Foundation of China (Grant No. 61075105) and the China Ministry of Education (Grant No.200802550007).

References

1. Galan, R.F., Fourcaud-Trocme, N., Ermentrout, G.B., Urban, N.N.: Correlation-induced synchronization of oscillations in olfactory bulb neurons. J. Neurosci. 26, 3646–3655 (2006)
2. Borgers, C., Kopell, N.: Synchronization in networks of excitatory and inhibitory neurons with sparse, random connectivit. Neural Comput. 15, 509–538 (2003)
3. Doiron, B., Chacron, M.J., Maler, L., Longtin, A., Bastian, J.: Inhibitory feedback required for network oscillatory responses to communication but not prey stimuli. Nature 421, 539–543 (2003)
4. Lindner, B., Doiron, B., Longtin, A.: Theory of oscillatory firing induced by spatially correlated noise and delayed inhibitory feedback. Phys. Rev. E. 72, 061919 (2005)
5. Marinazzo, D., Kappen, H.J., Gielen, S.C.A.M.: Input-driven oscillations in networks with excitatory and inhibitory neurons with dynamic synapses. Neural Comput. 19, 1739–1765 (2007)
6. Niessing, J., Ebisch, B., Schmidt, K.E., Niessing, M., Singer, W., Galuske, R.A.W.: Hemodynamic signals correlate tightly with synchronized gamma oscillations. Science 309, 948–951 (2005)
7. Hasenstaub, A., Shu, Y., Haider, B., Kraushaar, U., Duque, A., McCormick, D.A.: Inhibitory postsynaptic potentials carry synchronized frequency information in active cortical networks. Neuron 47, 423–435 (2005)
8. Bair, W., Zohary, E., Newsome, W.T.: Correlated firing in macaque visual area MT: Time scales and relationship to behavior. J. Neurosci. 21, 1676–1697 (2001)
9. Kohn, A., Smith, M.A.: Stimulus dependence of neuronal correlation in primary visual cortex of the macaque. J. Neurosci. 25, 3661–3673 (2005)
10. de La Rocha, J., Doiron, B., Shea-Brown, E., Josic, K., Reyes, A.: Correlation between neural spike trains increases with firing rate. Nature 448, 802–806 (2007)
11. Smith, M.A., Kohn, A.: Spatial and temporal scales of neuronal correlation in primary visual cortex. J. Neurosci. 28, 12591–12603 (2008)
12. Morita, K., Kalra, R., Aihara, K., Robinson, H.P.: Recurrent synaptic input and the timing of gamma-frequency-modulated firing of pyramidal cells during neocortical "UP" states. J. Neurosci. 28, 1871–1881 (2008)
13. Lytton, W.W., Sejnowski, T.J.: Simulations of cortical pyramidal neurons synchronized by inhibitory interneurons. J. Neurophysiol. 66, 1059–1079 (1991)
14. Xie, J., Wang, Z., Shi, H.: Non-monotonic relationship between correlation and strength of inhibitory feedback. In: 2010 Third International Conference on Modelling and Simulation, Wuxi (2010)
15. Xie, J., Wang, Z., Shi, H.: Robust non-monotonic relationship between output correlation and strength of inhibitory feedback. In: The 2010 Sixth International Conference on Natural Computation, Yantai (2010)

A New Model to Simulate the Formation of Orientation Columns Map in Visual Cortex

Hui Wei and Yun Wang

School of Computer Science, Fudan University,
Zhangheng Road. 825, 201203, Shanghai, P.R.C. China
weihui@fudan.edu.cn, wwangyunn@gmail.com

Abstract. The new model presented in the paper is used to simulate the development process of the orientation selectivity in primary visual cortex. The model combines mechanisms such as receptive field control, lateral connections, function columns into a network and then trained with random samples, can be regarded as a first stone of other feature maps. The model attempts to verify the basic features of the orientation map such as singularities, continuity and diversity. Meanwhile the model can be expanded with increasing columns or hyper-columns easily in order to process lager scope of stimulus. Another point is fault-tolerance, if some column is not successfully trained, the map can still perform well. After fast training process, the image of finished orientation map displaying with topology function is similar with the biological cortex orientation map, and the formed map can be used to extra the orientation information of the input quickly for the further visual process.

Keywords: development, primary visual cortex, orientation selectivity, orientation map.

1 Introduction

The primary visual cortex, which is also known as V1 or area 17, can be view as a topographic map like many other areas of the neo-cortex. Adjacent neurons in it respond to near regions of the receptive field. Neurons in primary visual cortex are responsive to certain features in the input, such as lines of a certain orientation, different colors, and movements of various directions. Orientation selectivity and ocular dominance are most thoroughly studied parts of the primary visual cortex, scientists from mathematics, neuroscience, biophysical and computer science, over the last several decades, have done thousands of experiments to study the neural mechanisms[1], trying various methods to dig out the features or producing a model to describe the formation of the orientation map.

In visual cortex, all neurons arranged in a vertical column and respond to stimuli oriented at the same angle is called orientation column. They typically have the same orientation preference. Neurons in a neighbor column will have a slightly different orientation preference, which gradually vary across the surface of the cortex that behaves the continuity of the organization of orientation columns [2, 3].

© Springer-Verlag Berlin Heidelberg 2011

Visual experience plays important role in cortical map, especially orientation map while altering the visual environment in an early stage would greatly change the organization of orientation column. If an animal is raised with both eyes closed, that is to say no stimulus is given to the cortex, the orientation columns do not form. If the animal is given only special orientation stimulus, the formed orientation map would only respond to the stimulus of this kind. This factor suggests that study efficiency of the model will get decrease as time goes on.

Neurons in orientation column have lateral connections to surrounding neurons. In some model, it seems like orientation column connect to neighboring columns laterally. Different models with different radius of lateral connections accomplish different task.

Computational models have been realized in the computer system, most of which are based on biological theory or experiments. Our method obey the classical principles and theory illustrated above and pay attention to the computational efficiency while not bring the model to a computer vision one.

Our approach has several differences from following computer vision functions: (1) though PCA, ICA can simulate orientation specificity, reconstruction, it cannot simulate other features of V1, such as continuous distribution mentioned above; (2) Theoretically, neurons in the network of PCA are completely connected, not meet that real receptive field roughly only has local-connections, and it wastes too many resources in training and cutting of connections in global connection matrix; (3) The geometric methods use Eigen-vectors as mask has a defect that resolution cannot change while some slight difference of image may lead to a great change in parameters; (4) functions aim at compression storage, transmission and entire reconstruction other than understanding the information in the view; (5) Rule of complex geometry such as Gestalt features is not easy to develop; (6) Real-life situation is continuous, which has spatial correlation in it.

2 Material and Method

Our model aims to simulate the formation of the orientation maps which obeys the principles of continuity, diversity, global disorder, singularities and linear zones [4] while also have the distinguish advantages such as ease of expansion; fast training for it have much less connections than other competitive network;

Previous models use various kinds of inputs [2] such as: elongated patches, initially uncritical, oriented stimulation, uncorrelated noise, natural images, radially uniform correlations, negative correlations between ON and OFF inputs. Here our model extra binary stimulus, pixels matrix from input after retina and LGN process, which distinguishes our model from others.

We use the ‘columns’ in the present paper, compared with the definition by Hubel and Wiesel (1974b), not limit the size of their receptive field, for different animals, cats, macaques have different ones. Its size roughly corresponds to the size of the cortical point image [2], not too small, not too big. Receptive field of neighboring columns overlap a lot, big proportion of the region, which help the formation of singular and continuity.

Assume that the whole output layer of the model make up a piece of the orientation map. The output layer should consist of tens to hundreds of hyper-columns. Besides cortical activation is under synchronous mechanism and competitive Hebbian models [6, 7], there is feedback system that the connections weaken slower than being enhanced.

Then here we will discuss details about the model we proposed on these basic.

Model:

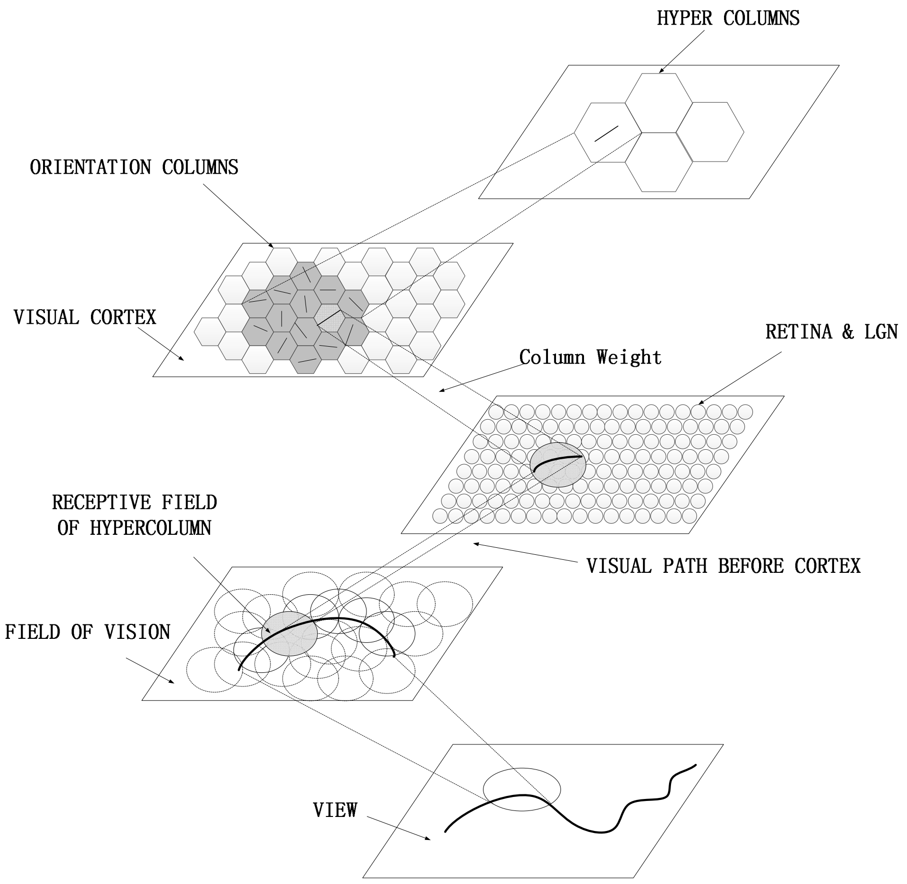


Fig. 1. Different neurons have Receptive Fields of different size. There are lots of neurons in retina layer whose receptive fields overlap his neighbors'. Through retina connection, each neuron in retina layer 'sees' the input after simple threshold processing. Through trained column weight, one column in hyper column will be selected to represent it. The columns in one region (gray region in the Figure.1) have the same receptive field in the view, that is to say, the same size and 'see' same thing at one time.

For each hyper column in map:

$$\mathrm{W_{t+1}(r) = W_t(r) + \alpha\beta\sigma(r,r')[v_{t+1} - W_t(r)]} \tag{1}$$

$$\sigma(\mathrm{r,r'}) = \exp(-|\mathrm{r - r'}|^2) \tag{2}$$

$$\mathrm{r'(t)} = \alpha \mathrm{R} \tag{3}$$

$$\beta = \eta + 1 - \eta[\mathrm{v_{t+1} - W_t(r)}]/|\mathrm{v_{t+1} - W_t(r)}| \tag{4}$$

W is weight of column; t is time or iteration times; α is study efficiency, which will decrease with t increases; v is the input vector from samples; $\sigma(\mathrm{r,r'})$ is a function about the distance from current neuron to the winner neuron. Neighbor radius will also decrease with t increases.

Another distinguish of the model is that the efficiency will decrease when $\mathrm{v_{t+1} - W_t(r) < 0}$ compared with the forgetting curve.

The present model not only obeys the principles of continuity and diversity but also contains singularities in the orientation preference map [1]. The orientation selectivity similarity is enhanced by altering weight vector of columns both in the same group and not, if only inside the neighboring radius, which moving them all closer to a presented input vector. Competition, implemented as a selection rule in model leads to diversity is the output map.

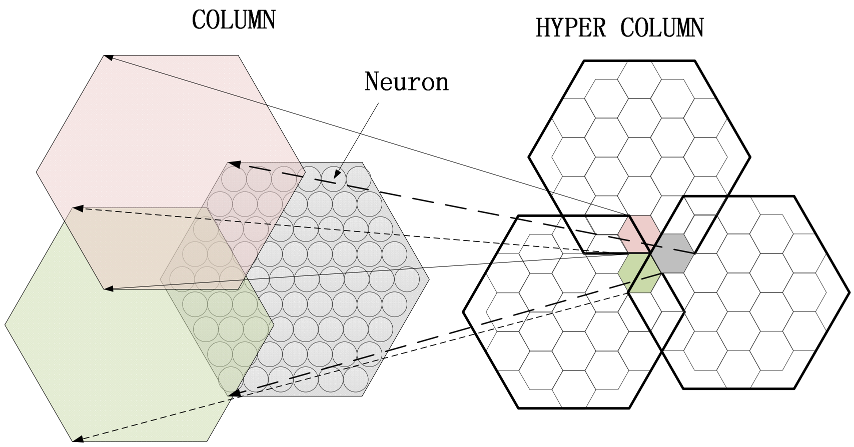


Fig. 2. Columns arranged in hexagon grid composed hyper column. Columns in one hyper columns share the same region in receptive field while they have different orientation selectivity from 0° to 180°. The angle varies continuously in the hyper column in most cases; Columns in neighboring hyper-columns share great proportion of region of receptive field.

Orientation map in the model does not place columns by the liquid-like properties; it is simply arranged in a crystal-like grid of exactly repeating placing hyper-columns.

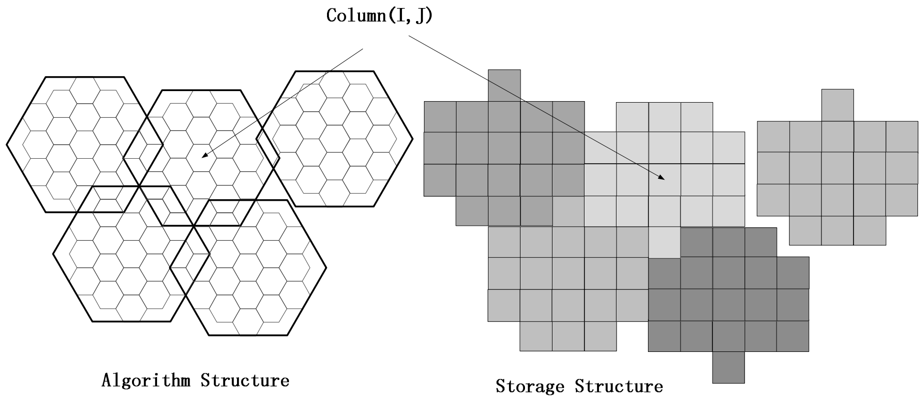


Fig. 3. The orientation map is stored in column level which we can regard as columns map. As shown in Figure 3, each hexagon-like column in displaying map (left: Algorithm Structure) has a corresponding column in storage map (right: Storage Structure); the whole hyper-column is similar to a Tetris of N bricks (N=19 on this occasion).

As shown in Figure 3, the storage map is easy to expand with such patch-like Tetris while it is easy to be stored with the simple data structure ‘array’. So our model has feature of easy expansion. It is of great weight when we change the experiment environment to a system of distributed architecture.

Supposed that central coordinate of column (I, J) in Algorithm Structure (Fig.3. left) is (X_a, Y_a); while central coordinate of column (I, J) in Storage Structure (Fig.3. right) is (X_s, Y_s).

$$X_a = Y_s \times \sin\theta \times \text{edge} \tag{5}$$

$$Y_a = X_s \times 2\cos\theta \times \text{edge} + \frac{Y_s\%2}{2} * \text{edge} \tag{6}$$

$\theta=\pi/6$, edge is the edge length of the hexagon in Figure.3. left. Edge would better be larger than number of receptive field neurons.

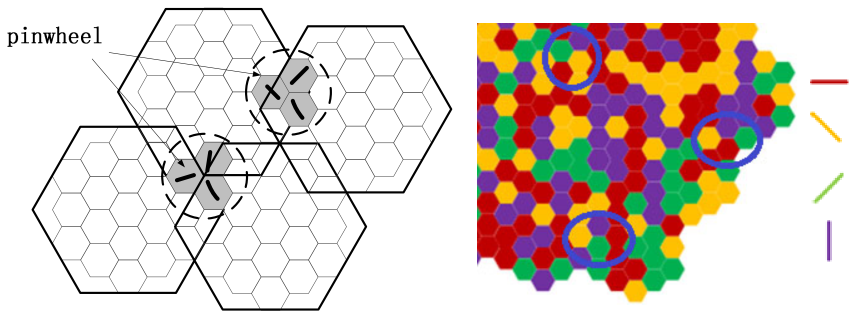


Fig. 4. The feature of singularity is the very important one, point-like discontinuities in the orientation map, around which orientation preferences increase/decrease clockwise. Figure on the left shows why the present model can form pinwheel or to say the model has feature of singularity. Hyper columns in near place share a common area, after long period learning, there has a chance to be reflected to a stimulus in the area; and then develop into a pinwheel.

Result:

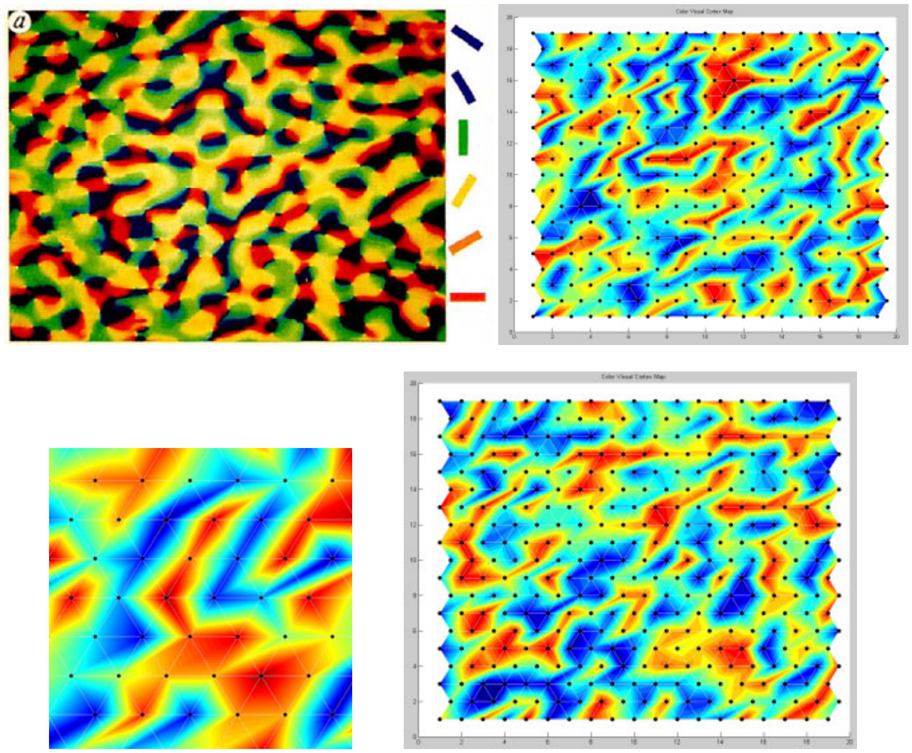


Fig. 5. A part of the orientation map produced by our model (bottom left) and the whole output (top right, bottom right) show the final result of the training function. Compared with the real orientation map with voltage-sensitive dyes ([8], top left), it is somehow similar. In fact the output map contains only 19*19 neurons; the difference from the map illustrated above is that we choose a topological display function with better performance in the whole view display. We can see that columns of different orientation selectivity are arranged averagely in the map. Though we cannot tell exact where there is a hexagon like hyper-column, we can easily find nearly columns of different orientations in a certain region. The principle of continuity and diversity, singularity and linear zone is represented well in the result.

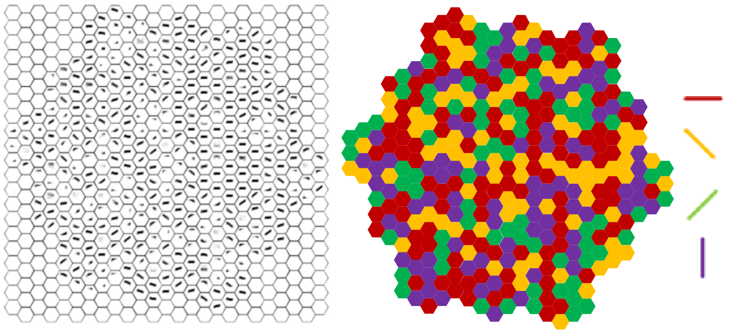


Fig. 6. The orientation columns map (on the right) is the result orientation map (on the left) represented with four color, color of each column represents orientation selectivity such as 0°, 45°, 90°, 135°. Though orientation selectivity of some columns is not exactly in these four, the model will use the most similar one to replace it.

We can find linear zones and singularity, pin-wheel like regions in the orientation map. And in the whole map the principle of continuity and diversity can also been seen.

Acknowledgment

This work was supported by 973 Program (Project No. 2010CB327900) and NSFC major project (Project No. 30990263).

References

1. Erwin, E., Obermayer, K., Schulten, K.: Models of orientation and ocular dominance columns in the visual cortex: a critical comparison. Neural Computation 7, 425–468 (1995)
2. Swindale, N.V.: The development of topography in the visual cortex: a review of models. Network 7(2), 161–247 (1996)
3. Hubel, D.G., Wiesel, T.N.: Sequence regularity and geometry of orientation columns in the monkey striate cortex. J. Comp. Neurol. 158, 267–293 (1974)
4. Shmuel, A., Grinvald, A.: Coexistence of linear zones and pinwheels within orientation maps in cat visual cortex. Proc. Natl. Acad. Sci. USA 97, 5568–5573 (2000)
5. Stemmler, M., Usher, M., Niebur, E.: Lateral interactions in primary visual cortex: a model bridging physiology and psychophysics. Science 269, 1877–1880 (1995)
6. Miikkulainen, R., Bednar, J.A., Choe, Y., Sirosh, J.: Computational maps in the visual cortex. Springer, New York (2005)
7. Bednar, J.A., Miikkulainen, R.: Pattern-generator driven development in self-organizing models. In: Computational Neuroscience: Trends in Research, pp. 317–323. Plenum, New York (1998)
8. Blasdel, G., Salama, G.: Voltage-sensitive dyes reveal a modular organization in monkey striate cortex. Nature 321, 579–585 (1986)

Study on the Synchrony Intensity Threshold of Two Uncoupled Neurons under Different Currents' Stimulation

Yueping Peng

Communication Engineering Department,
Engineering College of Armed Police Force, 710086, Xi'an, China
Percy001@163.com

Abstract. The input current of two uncoupled Hindmarsh-Rose neurons under different initial conditions is modulated by the different membrane potential of the Hindmarsh-Rose neuron; and the synchrony intensity threshold of two uncoupled neurons under different currents' stimulation by calculating and analyzing their maximum absolute phase difference.Under different simulation signals, the two uncoupled neurons can realize the phase synchronization or the full synchronization, and the stimulation intensity threshold of the two uncoupled neurons' realizing synchronization is different. According to the signal's complexity, the more complex the stimulation signal is, the smaller its intensity threshold to realize the two uncoupled neurons' synchronization is. Under the chaos signal's stimulation, its intensity threshold to realize the two uncoupled neurons' synchronization is smaller than the period signal, and is easier than the period signal to realize the two uncoupled neurons' synchronization. So the chaos discharge paterns is more favourable to signals' expression and transmission in neural system. From the calcium ion's effect, the smaller the stimulation neuron's parameter *r* is, the smaller the effect of the stimulation signal's calcium ion is, the easier the two uncoupled neurons realize synchronization. So the stimulation signal whose calcium ion's effect is large isn't easy to realize the two uncouple neurons' synchronization. This investigation shows the synchrony intensity threshold's rule of two uncoupled neurons under different currents' stimulation. These results are helpful to study synchronization and encode of many neurons or neural network.

Keywords: The Hindmarsh-Rose neuron; Synchronization; Threshold.

1 Introduction

Since system synchronization was presented by Pecora, et al[1, 2] in 1991, synchronization research has been causing researchers' wide focus in neuroscience field. Nervous activities' synchronization is found not only among coupled neuron groups in the same brain region, but also among uncoupled neuron groups in the same brain region or among different cortical areas; Moreover it can cross over two semispheres of the brain[3]. So in the nervous system, synchronization activities are presented not only among the coupled neurons, but also among the uncoupled

© Springer-Verlag Berlin Heidelberg 2011

neurons. Studies on neuron synchronization are mainly focused on two cases: the coupled neurons and the uncoupled neurons; And the synchronization of coupled neurons is studied more.

In 2002, A.B.Neiman confirmed that noise can realize the uncoupled sensory neurons' synchronization in the experiment for the first time[4]. Later, some researchers also discussed the synchronization problem of the uncoupled neurons[5-12]. But the synchronization mechanism of the uncoupled neurons isn't clear yet; For example, the synchrony intensity threshold of two uncoupled neurons is often much large, which can't be explained in sound reasons.

The Hindmarsh-Rose neuron(HR neuron) has several characteristics of the excitable cell's physical model and many time scale discharge action[13-15, 20]; There are some literatures of studies on synchronization of HR neurons[3, 5, 11, 16, 19, 21-23]. Huerta R and Rabinovich M I studied the period rhythm's change of two HR neurons coupled by the circuit and the synapse[16]. Two uncoupled HR neurons can be realized synchronization by applying input signals to modulate the neuron model's parameter or by applying the noise and the HR neuron's membrane potential to stimulate these neurons[5, 11, 17]. The phase synchronization of two coupled HR neurons was discussed by Jian-Wei Shuai and Durand D M, and they concluded that the phase synchronization is the discharge synchronization, the frequency synchronization is the cluster synchronization, and the full synchronization is the state synchronization[18]. Synchrony of two uncoupled neurons under half wave sine current stimulation was discussed, and it was concluded that the two uncoupled HR neurons under different initial conditions, whose parameter *r* is different or the same, can realize discharge synchronization(phase synchronization) or the full synchronization(state synchronization)[19]. Above researches are not systematical and all-round. It is too monadic in the stimulation signals' choice, and it is not studied and analyzed systematically and by contrast in the synchrony intensity threshold under different currents' stimulation.

In this study, we take two uncoupled HR neurons as the object, and make the HR neurons different initial discharge patterns by setting the value of the parameter *r*, and apply the different membrane potentials of the Hindmarsh-Rose neuron to modulate the two uncoupled HR neurons' input current, and discuss the synchrony intensity threshold of two uncoupled neurons under different currents' stimulation by calculating and analyzing these two neurons' membrane potentials and their maximum absolute phase difference.

2 The Discharge Patterns of the HR Neuron Model

The HR neuron has many time scale dynamics action, and its equation is set of three dimension ordinary nondimensional differential equations[13-15],

$$\begin{aligned} \dot{x} &= y - ax^3 + bx^2 - z + I \\ \dot{y} &= c - dx^2 - y \\ \dot{z} &= r[s(x - X) - z] \end{aligned} \tag{1}$$

The HR neuron has three time variables: the membrane potential variable *x* which has the quick depolarization ability, the quick recovery variable *y* and slow adaptation current variable *z*. *I* is input stimulation current; *a*, *b*, *c*, *d*, *s*, *r* and *X* are parameters. The parameter *r* is related to the membrane penetration of calcium ion, and reflects the changing speed of the slow adaptation current variable *z*. Other parameters have no specific physical meaning. The equations are nondimensional, and at numerical calculation, value of parameters is as follows: *a*=1.0, *b*=3.0, *c*=1.0, *d*=5.0, *s*=4.0, *I*=3.0, *X*=-1.56, and you can make the neuron different discharge patterns by controlling the parameter *r* variation.

The discharge threshold value is -0.25, and if the membrane potential is more than -0.25, the neuron will produce one discharge process. Fig.1 is the ISI bifurcation figure of the neuron, where parameters except *r* are set to the above values, and the initial state of the neuron is (1.0, 0.2, 0.2). From Fig.1, the discharge pattern of the neuron begins from the chaos state(*r* is about 0.008~0.009), and evolves period 6 discharge pattern(*r* is near 0.01), and via the adverse period doubling bifurcation passes period 3(*r* is about 0.0105~0.012) and enters the chaos state(*r* is about 0.0125~0.015) again, and at last via the adverse period doubling bifurcation passes period 4(*r* is about 0.016~0.018) and comes into period 2(*r* is about 0.0185 ~0.022).

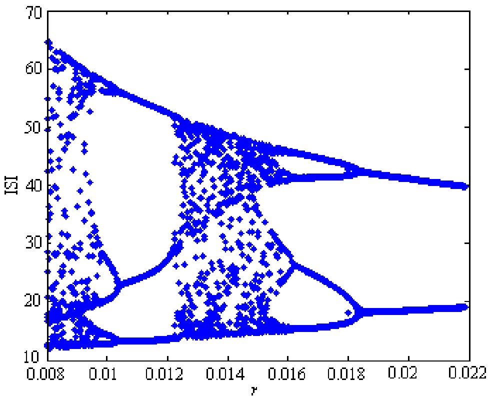


Fig. 1. Bifurcation figure of the HR neuron under the parameter *r* changing from 0.008 to 0.022

3 Study on the Synchrony Intensity Threshold of Two Uncoupled Neurons under Different Currents' Stimulation

The equation set of two uncoupled HR neurons' model is:

$$\dot{x}_i = y_i - a_i x_i^3 + b_i x_i^2 - z_i + (I_i + I_S(t))$$

$$\dot{y}_i = c_i - d_i x_i^2 - y_i$$

$$\dot{z}_i = r_i[s_i(x_i - X_i) - z_i] \quad (i=1,2) \tag{2}$$

Time variables of these two neurons are respectively (x_1, y_1, z_1) and (x_2, y_2, z_2),, and the parameters of these two neurons are respectively (a_1, b_1, c_1, d_1, r_1, s_1, I_1, X_1) and (a_2, b_2, c_2, d_2, r_2, s_2, I_2, X_2).

The equations are also nondimensional. At numerical calculation, the values of these two neurons' parameters (except r) are as follows: a_1=a_2=1.0, b_1=b_2=3.0, c_1=c_2=1.0, d_1=d_2=5.0, s_1=s_2=4.0, X_1=X_2=-1.56, I_1=I_2=3.0; And the values of the initial states of these two neurons are respectively (1.0, 0.2, 0.2) and (-1.0, 0.8, 0.3); And you can set these two neurons at different discharge patterns by controlling values of the parameter r_i (i=1, 2).

According to the difference of these two neurons' parameter *r*, there are three cases: the parameter *r* is the same, a little different($|r_1 - r_2| \le 0.0005$), and much different($|r_1 - r_2| \ge 0.003$).

The value of the initial state of the stimulation neuron is (0.2, 1, -0.2). The values of the parameters except *r* of the stimulation neuron are as follows: *a*=1.0, *b*=3.0, *c*=1.0, *d*=5.0, *s*=4.0, *I*=3.0, *X*=-1.56. The stimulation neuron can be made different discharge patterns by changing the parameter *r* according to Figure 1.

Total stimulation current includes two parts: the bias current(I_1 and I_2) and the input stimulation current I_S(t): $I_S(t) = kx(t)$. *X*(t) is the membrane potential of the stimulation neuron, and *k* is the stimulation strength of the membrane potential. The stimulation neuron begins to stimulate the system model after the bias current(I_1 and I_2) has been working for 500. The simulation time of the system model is often 0~4000.

3.1 The HR Neuron's Phase Function and the Synchronization Judge Rules

Synchronization state of two neurons can be classified by the phase function[18, 19]. The phase function of these two neurons is here defined as:

$$\phi_i(t) = \arctan[\dot{x}_i(t-0.5)/(\dot{x}_i(t)+0.1)] \quad (\text{i=1, 2}) \tag{3}$$

where x(t) is the membrane potential.

The absolute phase difference of two neurons($|\Delta\phi(t)|$) is defined as:

$$|\Delta\phi(t)| = |\phi_1(t) - \phi_2(t)| \tag{4}$$

Rules to judge two uncoupled HR Neurons' synchronization by the absolute phase difference of two neurons($|\Delta\phi(t)|$) are as follows:

When the maximum absolute phase difference of these two neurons ($|\Delta\phi(t)|_{\max}$) is more than 4π, these two neurons don't realize synchronization; when the maximum absolute phase difference of these two neurons ($|\Delta\phi(t)|_{\max}$) is no more than 4π for some small time intervals and be more than 2π, these two neurons can be viewed as an intermittent discharge synchronization; When the maximum absolute phase difference of these two neurons ($|\Delta\phi(t)|_{\max}$) is no more than 2π, these two neurons realize discharge synchronization(phase synchronization); When the maximum absolute phase difference of these two neurons ($\left|\Delta\phi(t)\right|_{\max}$) approximates to zero, these two neurons maybe realize the full synchronization.

Fig.2 shows the changing diagram of $\left|\Delta\phi(t)\right|_{\max}$ with the stimulation strength *k* changing. According to above rules to judge two uncoupled HR Neurons' synchronization by the maximum absolute phase difference of these two neurons ($\left|\Delta\phi(t)\right|_{\max}$), these following results can be concluded from Fig.2. When the stimulation strength *k* changes from 0 to 12 according to the step 0.02, these two neurons' discharge patterns begin with asynchronization, and gradually realize the discharge synchronization via the quick intermittent discharge synchronization process. When the stimulation strength *k* is near 2, and these two neurons in Fig.2.(a) go into the intermittent discharge synchronization state, so the synchrony intensity threshold of these two neurons is about 2; When the stimulation strength *k* is near 2.4, and these two neurons in Fig.2.(b) go into the intermittent discharge synchronization state, so the synchrony intensity threshold of these two neurons is about 2.4; When the stimulation strength *k* is near 3, and these two neurons in Fig.2.(c) go into the intermittent discharge synchronization state, so the synchrony intensity threshold of these two neurons is about 3.

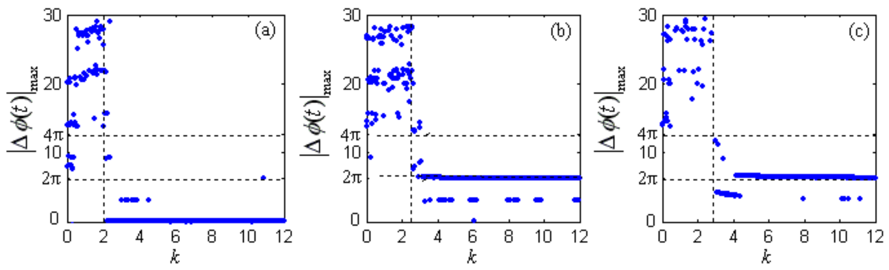


Fig. 2. The changing diagram of the maximum absolute phase difference of these two neurons with the stimulation strength k changing, where the parameter r of the stimulation neuron is 0.02, and the changing step of the stimulation strength k is 0.02. (a) The parameter r of these two neurons is the same: $r_1=r_2=0.013$. (b) The parameter r of these two neurons is a little different: $r_1=0.009$ and $r_2=0.01$. (c) The parameter r of these two neurons is much different: $r_1=0.014$ and $r_2=0.0085$.

3.2 The Synchrony Intensity Threshold of Two Uncoupled Neurons under Different Currents' Stimulation

Recent researches showed that when two uncoupled neurons realize synchronization under different signals' stimulation, the stimulation intensity must reach a certain threshold, and different stimulation signals have different synchrony intensity threshold[15, 19-23]. Brain is the chaos state at most time, and the neuron's research about the chaos control has been becoming the hot research field. So in the paper, the issue of the two uncoupled neurons' synchrony intensity threshold is discussed, and these neurons' initial discharge patterns are all the chaos.

The values of these two neurons' parameters (except *r*) are as above. According above the difference criterion of these two neurons' parameter *r*, Table 1 shows the values of the two neurons' parameter *r*, and the discharge patterns of these two neurons are correspondingly showed in the Fig.1. From Fig.1, the initial discharge patterns of these two neurons are all the chaos.

Table 1. These two neurons' parameter *r* and their initial discharge patterns

Case	The values of r_1 and r_2	The initial discharge pattern of neuron 1	The initial discharge pattern of neuron 2
The same	$r_1=r_2=0.009$	Chaos	Chaos
	$r_1=r_2=0.013$	Chaos	Chaos
A little different	$r_1=0.0085$, $r_2=0.009$	Chaos	Chaos
	$r_1=0.014$, $r_2=0.0141$	Chaos	Chaos
Much different	$r_1=0.014$, $r_2=0.0085$	Chaos	Chaos

The stimulation signals are the period signal and the chaos signal which is produced by the HR neuron by changing the parameter *r* according to Figure 1. The types of the stimulation signals are showed in the Table 2.

Table 2. The types of the stimulation signals and Its producing methods

types of the stimulation signals	producing methods
period 2	produced by the HR neuron whose parameter *r* is 0.02
period 4	produced by the HR neuron whose parameter *r* is 0.017
chaos	produced by the HR neuron whose parameter *r* is 0.014
chaos	produced by the HR neuron whose parameter *r* is 0.0085

Fig.3~Fig.6 shows the changing diagram of $|\Delta\phi(t)|_{\max}$ of the two uncoupled neurons whose parameter r are showed in the Table 1 with the stimulation strength *k* of different stimulation signals changing. In Fig.3~Fig.6, the stimulation signal is accordingly the period 2(*r* is 0.02), the period 4(*r* is 0.017), the chaos signal(*r* is 0.014), and the chaos signal(*r* is 0.0085), which are showed in Table 2.

From Fig.3~Fig.6, under the four simulation signals showed in Table 2, these two uncoupled neurons can't realize synchronization when the stimulation strength *k* is very small; With the stimulation strength *k'* increasing gradually, these two uncoupled neurons realize the discharge synchronization via the quick intermittent discharge synchronization process.

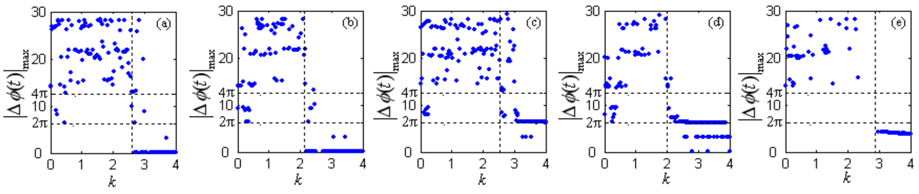


Fig. 3. The changing diagram of the maximum absolute phase difference of these two neurons with the stimulation strength k changing under the period 2 signal's stimulation(r=0.02). (a) r_1=r_2=0.009. (b) r_1=r_2=0.013. (c) r_1=0.0085, r_2=0.009. (d) r_1=0.014, r_2=0.0141.(e) r_1=0.014, r_2=0.0085.

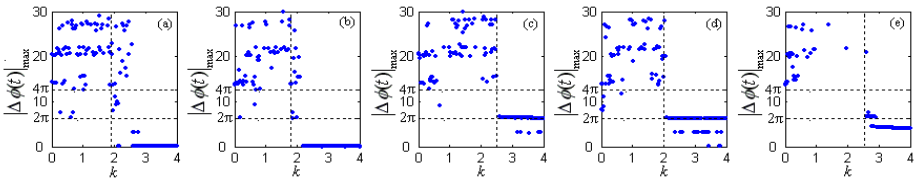


Fig. 4. The changing diagram of the maximum absolute phase difference of these two neurons with the stimulation strength k changing under the period 4 signal's stimulation(r=0.017). (a) r_1=r_2=0.009. (b) r_1=r_2=0.013. (c) r_1=0.0085, r_2=0.009. (d) r_1=0.014, r_2=0.0141.(e) r_1=0.014, r_2=0.0085.

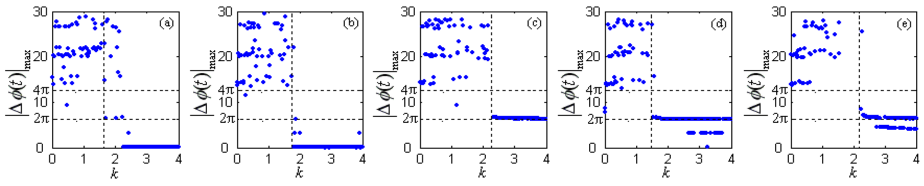


Fig. 5. The changing diagram of the maximum absolute phase difference of these two neurons with the stimulation strength k changing under the chaos signal's stimulation(r=0.014). (a) r_1=r_2=0.009. (b) r_1=r_2=0.013. (c) r_1=0.0085, r_2=0.009. (d) r_1=0.014, r_2=0.0141.(e) r_1=0.014, r_2=0.0085.

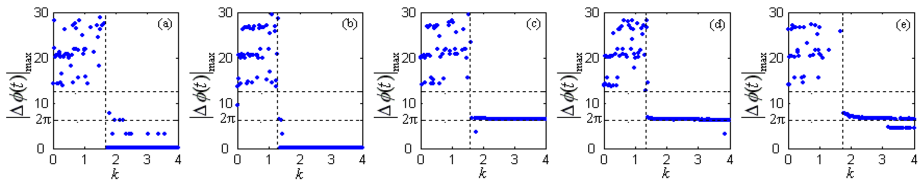


Fig. 6. The changing diagram of the maximum absolute phase difference of these two neurons with the stimulation strength k changing under the chaos signal's stimulation(r=0.0085). (a) r_1=r_2=0.009. (b) r_1=r_2=0.013. (c) r_1=0.0085, r_2=0.009. (d) r_1=0.014, r_2=0.0141.(e) r_1=0.014, r_2=0.0085.

From Fig.3~Fig.6, according to the stimulation intensity values of these neurons' going into the intermittent discharge synchronization state, the intensity threshold of these two uncoupled neurons realizing synchronization can be decided. Table 3 shows the stimulation intensity threshold of these two uncoupled neurons realizing synchronization under the four stimulation signals showed in the Table 2.

Table 3. The current stimulation intensity threshold of two uncoupled HR neurons realizing synchronization under the four stimulation signals showed in the Table 2

types of the stimulation signals	the stimulation intensity threshold				
	r_1=r_2=0.009	r_1=r_2=0.013	r_1=0.0085, r_2=0.009	r_1=0.014, r_2=0.0141	r_1=0.014 , r_2=0.0085
period 2 (r=0.02)	2.6	2.1	2.5	2	2.9
period 4 (r=0.017)	1.9	1.8	2.5	2	2.6
Chaos (r=0.014)	1.8	1.9	2.3	1.5	2.2
Chaos (r=0.0085)	1.7	1.3	1.6	1.4	1.8

From Table 3, under the different signals' stimulation, the stimulation intensity threshold of the two uncoupled neurons' realizing synchronization is different. According to the order from small to large, the synchronization stimulation intensity threshold is as following:

period 2(r=0.02) > period 4(r=0.017) > chaos(r=0.014) > chaos(r=0.0085)

According to the signal's complexity, the period 2 signal is the simplest and its synchronization stimulation intensity threshold is the largest; The chaos signal is complex and its synchronization stimulation intensity threshold is smaller. So, in general, the more complex the stimulation signal is, the smaller its intensity threshold to realize the two uncoupled neurons' synchronization is. Under the chaos signal's stimulation, its intensity threshold to realize the two uncoupled neurons' synchronization is smaller than the period signal, and is easier than the period signal to realize the two uncoupled neurons' synchronization; In other words, the chaos discharge paterns is more favourable to signals' expression and transmission in neural system.

In Table 2, according to the order from large to small, the value of the stimulation signal's parameter r is as following:

the period 2(r=0.02) > the period 4(r=0.017) > the chaos(r=0.014) > the chaos(r=0.0085)

The parameter r of the HR neuron is related to the membrane penetration of calcium ion, and reflects the changing speed of the slow adaptation current z. The larger the parameter r of the HR neuron is, and the faster the intramembranous calcium ion accumulates. However, from Table 3, the stimulation intensity threshold of two uncoupled HR neurons realizing synchronization is consistent with the value of the four stimulation signals' parameter r according to the order from large to small. So, the smaller the stimulation neuron's parameter r is, the smaller the effect of the stimulation signal's calcium ion is, the easier the two uncoupled neurons realize synchronization. But the stimulation signal whose calcium ion's effect is large isn't easy to realize the two uncouple neurons' synchronization.

4 Conclusion

Under different simulation signals, the two uncoupled neurons realize the phase synchronization or the full synchronization, and the stimulation intensity threshold of the two uncoupled neurons' realizing synchronization is different. According to the signal's complexity, the more complex the stimulation signal is, the smaller its intensity threshold to realize the two uncoupled neurons' synchronization is. Under the chaos signal's stimulation, its intensity threshold to realize the two uncoupled neurons' synchronization is smaller than the period signal, and is easier than the period signal to realize the two uncoupled neurons' synchronization. So the chaos discharge paterns is more favourable to signals' expression and transmission in neural system. From the calcium ion's effect, the smaller the stimulation neuron's parameter *r* is, the smaller the effect of the stimulation signal's calcium ion is, easier the two uncoupled neurons realize synchronization. So he stimulation signal whose calcium ion's effect is large isn't easy to realize the two uncouple neurons' synchronization.

This investigation shows the synchrony intensity threshold's rule of two uncoupled neurons under different currents' stimulation. These results are helpful to study synchronization and encode of many neurons or neural network.

Acknowledgments. This work is supported by the basic research project of Engineering College of Armed Police Force. The project name is "Study on the Synchronization Mechanism of Uncoupled Neurons (No. WJY201015).

References

1. Pecora, L.M., Carroll, T.L.: Synchronization in Chaotic Systems. Physical Review Letter 64, 821–824 (1990)
2. Pecora, L.M., Carroll, T.L.: Driving Systems with Chaotic Signals. Physical Review: A 44, 2374–2383 (1991)
3. Gray, C.M., et al.: Oscillatory Response in Cat Visual Cortex Exhibit Inter-columnar Synchronization Which Reflects Global Stimulus Properties. Nature 338, 334–337 (1989)
4. Neiman, A.B., Russell, D.F.: Synchronization of Noise-induced Bursts in Noncoupled Sensory Neurons. Phys. Rev. Lett. 88, 138103-1–138103-4 (2002)
5. Daihai, H., Pengliang, S., Stone, L.: Noise-induced Synchronization in Realistic Models. Physical Review: E 67, 0272011–0272013 (2002)
6. Yoshida, K., Sato, K., Sugamata, A.: Noise-induced Synchronization of Uncoupled Nonlinear Systems. Journal of Sound and Vibration 290, 34–47 (2006)
7. Xia, S., Qi-Shao, L.: Coherence Resonance and Synchronization of Hindmarsh-Rose Neurons with Noise. Chinese Physics 14, 1088–1094 (2005)
8. Wu, Y., Xu, J.-X., Jin, W.-Y., Hong, L.: Detection of Mechanism of Noise-induced Synchronization between Two Identical Uncoupled Neurons. Chin. Phys. Lett. 24, 3066–3069 (2007)

9. Wu, Y., Xu, J., He, D., Earn, D.J.D.: Generalized Synchronization Induced by Noise and Parameter Mismatching in Hindmarsh-Rose Neurons. Chaos, Solitons and Fractals 23, 1605–1611 (2005)
10. Wu, Y., Xu, J., He, D., Jin, W.: Study on Nonlinear Characteristic of Two Synchronizing Uncoupled Hindmarsh-Rose Neurons. Acta Physica Sinica 54, 3457–3464 (2005)
11. Wu, C., Xu, J., Jin, W.: Complete Synchronization and Phase Synchronization of Two Uncoupled Neurons through Parametrical drive. Journal of Xi'an Jiaotong University 39, 544–547 (2005)
12. Wu, Y., Xu, J.-X., He, D., Jin, W., He, M.: Synchronization in Two Uncoupled Chaotic Neurons. In: Yin, F.-L., Wang, J., Guo, C. (eds.) ISNN 2004. LNCS, vol. 3173, pp. 138–143. Springer, Heidelberg (2004)
13. Hindmarsh, J.L., Rose, R.M.: A Model of the Nerve Impulse Using Two First-order Differential Equation. Nature 296, 162–165 (1982)
14. Hindmarsh, J.L., Rose, R.M.: A Model of Neuronal Bursting Using Three Coupled First Order Differential Equations. Series B, Biological Sciences 221, 87–102 (1984)
15. Peng, Y., Jian, Z., Wang, J.: Study on Discharge Patterns of Hindmarsh-Rose Neurons Under Slow Wave Current Stimulation. In: Jiao, L., Wang, L., Gao, X.-b., Liu, J., Wu, F. (eds.) ICNC 2006. LNCS, vol. 4221, pp. 127–134. Springer, Heidelberg (2006)
16. Huerta, R., Rabinovich, M.I.: Spike-train Bifurcation in Two Coupled Chaotic Neurons. Physical Review: E 55, R2108–R2110 (1997)
17. Ying, W., et al.: Study on Nonlinear Characteristic of Two Synchronizing Uncoupled Hindmarsh-Rose neurons. Physics Letters: A 54, 3457–3464 (2005)
18. Shuai, J.-W., Durand, D.M.: Phase Synchronization in Two Coupled Chaotic Neurons. Physics Letters: A 264, 289–296 (1999)
19. Peng, Y., et al.: Synchrony of Two Uncoupled Neurons under Half Wave Sine Current Stimulation. Communications in Nonlinear Science and Numerical Simulation 14, 1570–1575 (2009)
20. Peng, Y., Wang, J.: Study on Synchrony of Two Uncoupled Neurons under the Neuron's Membrane Potential Stimulation. The Journal of Biomedical Science and Engineering 3, 160–166 (2010)
21. Peng, Y., Wang, J.: Synchrony of Two Uncoupled Neurons under the Chaos Signal Stimulation. In: The 2nd International Conference on Computer Modeling and Simulation, pp. 355–359 (2009)
22. Peng, Y., Wang, J.: Study on Synchrony of Two Uncoupled Neurons under Slow Ramp Current Stimulation. In: 2009 Fifth International Conference on Natural Computation, pp. 418–422 (2009)
23. Peng, Y., Wang, J.: Discharge Patterns' Change of Hindmarsh-Rose Neurons under Slow Ramp Current Stimulation. In: 2007 2nd International Conference on Bio-Inspired Computing: Theories and Applications, pp. 52–55 (2007)

Research on Design Method of Small World Property ESN

Yingchun Bo[1], Junfei Qiao[1], and Shuwei Wang[2]

[1] College of Electronic and Control Engineering, Beijing University of Technology, 100124, Beijing, China
boyingchun@sina.com, junfeiq@bjut.edu.cn

[2] College of Architecture and Civil Engineering, Beijing University of Technology, 100124, Beijing, China

Abstract. Aim to solve the problems of structure design and parameters selection about conventional ESN, a small world property ESN (SWESN) is proposed in this paper. Neuron space growth algorithm is adopted to generate a physical network with small world topology on 2-D plane firstly, and then the nodes and the connections of the physical network are mapped into the neurons in reservoir of SWESN, Thus the dynamic neuron reservoir (DNR) in SWESN has small world characteristic. In addition, different typical neurons are adopted in the reservoir. The simulation experiments confirms that the SWESN generated by this method could create more abundant dynamic behavior than conventional ESN, and SWESN exceeds conventional ESN both at robustness and at anti-disturbance ability.

Keywords: echo state network; small world; dynamic neurons reservoir; robustness.

1 Introduction

Echo state networks (ESN) was firstly proposed in [1]. The ESN model contains a completely random state reservoir as a hidden layer, which is usually composed of hundreds or thousands of internal neurons. The internal neurons contain information about the history of input and output patterns. The outputs of these internal neurons (echo states) are fed into a memory-less but adaptive readout network (generally linear) that produces the network output. Differing form previous techniques tune all synaptic connections, the promising property of ESN is that only the memory-less readout is trained, whereas the recurrent topology has fixed connection weights. This reduces the complexity of RNN training while preserving a recurrent topology. This very promising RNN partially reflects some features of learning mechanisms in biological brains [2]. ESN has been widely applied for system identification, chaotic time-series prediction, and modeling of nonlinear dynamic system [1–4]. Owing to the high accuracy of ESN in many classical tasks, ESN has been a hot research topic in recently years [3, 4].

However, conventional ESN has still some problems to overcome[5]. The central issue about solving this problem focuses on how to select an appropriately

© Springer-Verlag Berlin Heidelberg 2011

reservoir (mainly the spectrum radius $\rho(W)$ and the sparseness of the reservoir) in view of a specific problem. A lager $\rho(W)$ is generally supposed to be related to a reservoir with rich dynamics. However, even though each ESN has the same sparseness and spectral radius, the performance obtained vary greatly among different realizations[2]. To obtain a good result, a tedious tests to select the parameters of W is needed. Many methods have been investigated to tune the structure of the ESN topology in literatures. Xue [6] thought that the utterly random links of the inner neurons could be resulted in strong coupling among them, and this degrade the richness of the internal dynamics. Modular idea was adopted in [6] to decompose the whole ESN into several local ESNs, and lateral inhibition was used to link each local ESN. The aim of this method was to generate several different inner dynamics in each module, and this way succeeded in solving MSO problem. Ozturk et al.[7] proposed a dynamic reservoir design method with maximizing the *Renyi* entropy.

Biological studies have shown that biological brain is a complex network consisted of a large number of neurons linking by great majority nerve fibers, and it has small world characteristic [8]. The small world property is benefit to organism to process information rapidly in local and in global, to realize local function and to integrate global functions [9]. In addition, the structure of the biological neural network is obviously modular [10]], and each module has its specific neuron type. Therefore, this paper investigated a small world property ESN (SWESN) to improve the robustness and to enhance the anti-noise ability of traditional ESN, .

This paper is organized as follows: In section 2, we analyze the principle of ESN in brief. A design method of SWESN is described in detail in section 3. Simulations and some analysis are introduced in section 4, and we draw some conclusions in section 5.

2 Problem of ESN

The update of the reservoir state is expressed as:

$$x(k+1) = \mathbf{f}(W_{in}\mathbf{u}(k+1) + W\mathbf{x}(k) + W_b\mathbf{y}(k)) \tag{1}$$

where $\mathbf{f}$ are activated functions, $\mathbf{x}$ are the internal states, $\mathbf{u}$ is the current input vector. and $\mathbf{y}$ is the action potential of the output neuron at the current time step, and $W_{in} \in \mathrm{R}^{N\times K}$, $W \in\mathrm{R}^{N\times N}$ and $W_b \in\mathrm{R}^{N\times L}$ are the weight matrices for input connections, the reservoir, and feedback connections, respectively. Where, K, N, L are the dimensions of the inputs, , the internal neurons and the outputs, respectively. The output of the ESN is:

$$y(k+1) = W_o\mathbf{x}(k+1) \tag{2}$$

The target of ESN training is to minimize the performance index J.

$$J = \frac{1}{2}\sum_{k=1}^{N_s}(d(k) - y(k))^2 \tag{3}$$

With N_s being the number of training samples, $d(k), y(k)$ being the expected output and the real output of ESN at time step k, respectively. Set $D = [d_1, , d_{N_s}]^T$, then,

$$W_o = \mathbf{X}^+ D \tag{4}$$

Where, X is the output sequence matrix of the internal neurons in reservoir, and X^+ is the Moore Penrose inverse of X. $X^+ = (X^T X)^{-1} X$. Obviously, if $X^T X$ is singular, then equation (4) would be ill-posed. In traditional ESN, $X^T X$ has some very small singular values, therefore, the solution of the equation (4) is hardly to be stable, and an effective method to alleviate this problem is to weaken the correlation of the output sequences of the internal neurons. The small world characteristic decomposes integrated neurons into several groups naturally, the links among groups are very sparse. If each group has its own specific spectrum radius or activated functions, the correlation among output sequences of the internal neurons would be weaker than that of conventional ESN. The SWESN would generate richer dynamics than ESN under the same number of the internal neurons. From function approximation view, it also means stronger approximation ability if the correlations among output sequences of the hidden-layer neurons trail off.

3 Design of SWESN

3.1 Generating Small World Networks

A kind of small world network generating method based on spatial increasing algorithm was proposed in [11]. The cortical systems networks development of macaque and cat were simulated by this way. And this method is very simple and effective. Firstly, a random point in 2-D plane is selected as an initial node, then, the following nodes link the existed nodes according to the probability calculated by the relation:

$$P(u,v) = \beta e^{-\alpha d(u,v)} \tag{5}$$

Where, $P(u,v)$ is the linking probability between nodes u and v, and $d(u,v)$ is the distance between both nodes u and v, and α, β are scaling coefficients. Apparently, the connection probability was dropped exponentially with the distance between u and v. So the connections among the nodes with small distances are much more than those with large distances. This method can build a network with small world property, in case α and β being selected properly [11].

However, this simple method to generate small world network has some disadvantages. The first is that the group number of the network can hardly be controlled. Second, bidirectional connections between nodes u and v can not be implemented by this method. Finally, this method can not realize the self-connection of neuron . Therefore, a little small change was adopted to solve these problems. The realizing approaches are as follows:

Step 1: Restrict the coordinate (x, y) of node in 2-D plane in a certain area($x \in (0,1)$ and $y \in (0,1)$ in this paper). Place N_c backbone nodes on the axis line determined by $(0,1)$ and $(1,0)$. The coordinates of the ith backbone node are $x_i = i/(N_c+1)$,$y_i = 1-i/(N_c+1)$. The backbone nodes will guide the following nodes to cluster around them, therefore,N_c groups of neurons will be generated along this way. The N_c backbone nodes are bidirectionaly full linked each other, and this network consisted of backbone nodes is the initial network. The number of nodes and the linking amount of the initial network are N_c and $N_c(N_c-1)$, respectively.

Step 2: Based on the initial network, the new added node connects the existed nodes with the probability obtained by equation (5), at the same time, all the existed nodes also connect the new added node with the same probability. Thus the bidirectional connections are created by this method. Noting that the connection between u and v (described as $l(u,v)$) is different from the connection between v and u (described as $l(v,u)$).

Step 3: After the network being created, each node can link itself with the probability P_s, and P_s is the self-connecting probability.

Further, we construct a matrix W_s, and the elements $W_s(i,j) = 1$ if there exists connection between node i and node j, otherwise $W_s(i,j) = 0$. Obviously, the nodes information and connections information can be reflected in W_s matrix. The coordinates of nodes are saved in NI matrix, and the connecting information among nodes are saved in LI matrix, which concludes the start node and the end node of each link.

3.2 Mapping the Nodes of Small World Network to the Neurons of SWESN

The performance of ESN is mainly determined by the structure of DNR. The connecting information of DNR is reflected in the reservoir matrix W. For analyzing simply, we rewrite the form of reservoir matrix as follows:

$$W = \begin{bmatrix} W_1 & & L_1 \\ & ... & \\ L_2 & & W_{N_c} \end{bmatrix} \tag{6}$$

Where, $W_c(c = 1, , N_c)$ is the connection matrix of the cth group, and L_1, L_2 are connecting matrices among different groups. Owing to the connections among different groups are extremely sparse, the non-zero elements in L_1, L_2 are very few. Though the W_s constructed in previous part is very similarly with W, there are also two key differences between them. First, the neuron ID in ith group is apparently continuous, while the serial numbers of the nodes in the same group in physical network generated in 2-D plane maybe not continuous, because the serial numbers of nodes in physical network is ordered by the sequence of nodes emerging, therefore, the serial numbers of nodes in W_s are not the same as that of neurons in W. Second, the element values in W_s (only 1 or 0) is apparently not corresponding with that in W. In order to realize the map between W_s and W, the following steps is adopted.

Step 1: Selecting the coordinate (0, 1) as the benchmark point, and then calculate the Euclidean distance $d_i(i = 1, ..., N_{max})$ between the benchmark point and the coordinate of the ith node, where N_max is the top limit of the neuron number in reservoir.

Step 2: Arrange d_i by ascending order, and then turn the serial number of node with the distance d_i into i. Finally, update the LI and NI matrices according to new node serial numbers.

After the rearrangement of nodes, the distances among serial numbers of the nodes are corresponding to the distances among coordinates of the nodes. And the new serial numbers of nodes can be served as the IDs of neurons in W matrix.

In SWESN, each group may contain different amount of neurons, i.e. the number of row and column maybe not same among different W_i, so we should discriminate which group each neuron subjects to. Apparently, the backbone nodes can be served as the center of the groups, therefore, the group that the neuron belongs to can be determined by

$$C_i = \arg(\text{c}) \min(d(NI_i, Z_c)) \tag{7}$$

Where, C_i is the ID of group that the ith neuron belong to, Z_c is the coordinate of the cth $(c = 1, , N_c)$ backbone neuron, and $d(NI_i, Z_c)$ is the Euclidean distance between the ith neuron and the cth backbone neuron. According to equation (7), the ith neuron belongs to the group with the nearest distance from the ith neuron to the group center.

The linking intensity among neurons are reflected by the element values of W matrix. There are three conditions in neurons connection: the first is the self-connection of each neuron, the second is the connections in the same group, and the last is the connections among groups. The three connection situations are denoted by S_{self}, S_{in}, and S_{out}, respectively.

Obviously, $LI_k \in S_{self}$, if $NI_i = NI_j$, $LI_k \in S_{self}$, if $NI_i \neq NI_j$ and $C_i = C_j$, $LI_k \in S_{out}$, if $NI_i \neq NI_j$ and $C_i \neq C_j$. Where, NI_i, NI_j are the coordinates of the ith neuron and the jth neuron, respectively. C_i and C_j are severally the groups of the ith neuron and the jth neuron belong to. $LI_k(i, j)$ denotes the kth connection (from the ith neuron to the jth neuron). The linking intensity among neurons can be determined by

$$W_{ij} = \begin{cases} ps - (1-p)s, \ if \ \ LI_k \in S_{self} \\ pa_c - (1-p)a_c, \ if \ \ LI_k \in S_{in} \text{ and } C_i = c \\ pb - (1-p)b, \ if \ \ LI_k \in S_{out} \end{cases} \tag{8}$$

Where, $\mathbf{a} = [a_1, ..., a_{N_c}]$, b, s are the intensity values ready for being selected by S_{in}, S_{self}, and S_{out} respectively, and a_c denotes the intensity values ready for the W_c, thus different spectrum radius can be generated with different groups. The dynamics in SWESN would be enhanced through selecting $\mathbf{a}, b$ and s values properly.

4 Simulations

A quantitative measure of robustness, termed the successful design ratio, is defined in equation (9). Where K is the number of Monte Carlo (MC) simulations, e_i is the generalization/prediction MSE for the ith MC simulation, $u(x)$ is the unit-step function, that is, $u(x) = 1$ when $x \leq 0$ and $u(x) = 0$ when $x > 0$. In words, $R(\theta)$ is an estimation of the probability of obtaining a network with generalization/prediction MSE equal to or less than the threshold θ.

$$R(\theta) = \frac{\sum_{i=1}^{K}(u(e_i - \theta)}{K} \tag{9}$$

We select MSO problem as research object. The MSO problem can be described as $y(k) = sin0.2k + sin0.311k$. The output y of MSO problem is the combination of two sine waves with different frequency. We generate 1000 time discrete sample points. The first 700 sample points are used to train the neural network, and the other 300 sample points are used to test. The internal states of the ESN and the SWESN are initialized randomly at the interval $[-1, 1]$. The number of the internal neurons is 200, and $\theta = 0.0034$ [6].

In order to investigate the relationships between the spectrum radius and robustness, in this experiment, we carry out 50 MC simulations at each spectrum radius, and totally select 20 different spectrum radiuses to test (the total number of MC simulations is 1000). The robustness of SWESN and ESN under different spectrum radius is shown in figure 1. In figure 1, the robustness index $R(\theta)$ is increased along with ρ roughly in total trend, either in ESN or in SWESN. The test MSE of SWESN is a little smaller than that of ESN almost in every spectrum radius (see figure 2), in addition, test MSE drop with the increase of spectrum radius to this general trend.

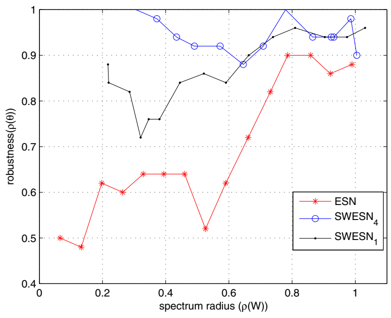


Fig. 1. Robustness circles of ESN and SWESN under different $\rho(W)$

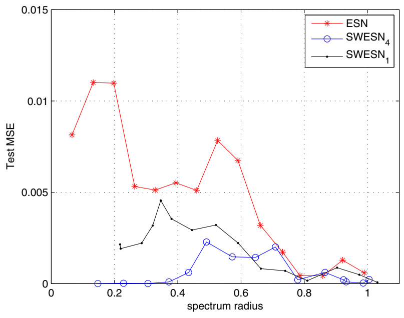


Fig. 2. Test MSE circles of ESN and SWESN under different $\rho(W)$

In fig.1 and fig.2, the subscript i in SWESN$_i$ denotes there is total i kinds of different activated functions are adopted in SWESN.

The singular value distribution is shown in table 1. There are more than 94% singular values are under 1.4×10^{-4} both to ESN and to SWESN. Remarkably, the percentage of singular value below 2.5×10^{-8} drops to 0 in SWESN, however it keeps a high degree (42.4%) in ESN.

Table 1. Singular value distribution in ESN and SWESN

singular value	ESN	SWESN
$\leq 1.4 \times 10^{-4}$	94.9%	94.2%
$\leq 3.1 \times 10^{-6}$	51.2%	24.2%
$\leq 2.5 \times 10^{-8}$	42.4%	0

There is always with noise in real data, so we select the real sun spot data set [12] to carry out the time-series prediction to investigate the anti-noise ability of the SWESN. There are total 3141 data in this set, the beginning 2000 data are selected, and the first 1000 sample points are applied to train the network, while the last 1000 sample points are applied to test. We execute 10-step prediction in this paper, i.e. for the current time step k, the output of the network is $y(k+10)$. The prediction circle is shown in figure 3. The relative error and sample point distribution of sun-spot prediction is shown in table 2. The predictively relative error less than 10% is about 95.1%, and that less than 5% is about 86.3% with SWESN, while in traditional ESN, both data are 87.6% and 74.8% respectively. Considering that we are carrying out 10-step prediction, the performance of SWESN is relatively better than that of traditional ESN.

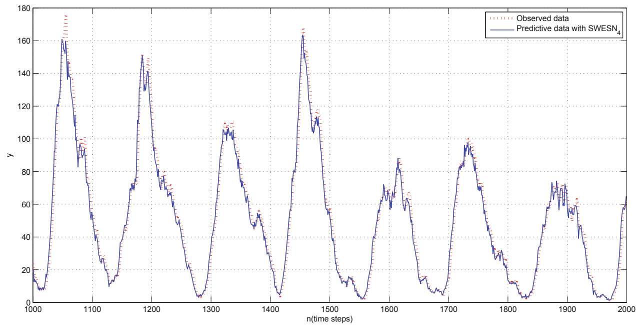


Fig. 3. The sun spot prediction circle of SWESN

Table 2. Relative error and sample point distribution of sun-spot prediction

relative error	≤1%	≤2%	≤5%	≤10%
SWESN	252	538	863	951
Conventional ESN	269	452	748	876

5 Conclusion

Inspired by the structure of cortical systems networks, this paper investigates a small world property ESN (SWESN). Compare to the traditional ESN, SWESN has some advantages. The first is that the connecting method with small world property degrades the coupling intensity among neurons in dynamic neuron reservoir, and enriches the dynamics of internal neurons, therefore the parameter adaptability of SWESN is better than that of traditional ESN. The second is that SWESN reduces the amount of very small singular value of $X^T X$, both the robustness and precision of generalization are increased. The last is that SWESN divides the whole ESN into several parts naturally, and the parameters can be tuned more flexible.

Acknowledgements. This study is supported by the National Natural Science Foundation of China under Grant No.61034008 and No.60873043; the National 863 Scheme Foundation of China under Grant No. 2007AA04Z160, the Doctoral Fund of Ministry of Education of China under Grant No. 200800050004; the Beijing Municipal Natural Science Foundation under Grant No. 4092010.

References

1. Jaeger, H.: The echo state" approach to analyzing and training recurrent neural networks. GMD Report. German National Research Center for Information Technology (2001)
2. Ozturk, M.C., Xu, D., Principe, J.C.: Analysis and Design of Echo State Networks. Neural Computation 19, 111–138 (2007)
3. Jaeger, H.: Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. Science 304, 78–80 (2004)
4. Liu, Y., Zhao, J., Wang, W.: Improved Echo State Network Based on Data-driven and Its Application to Prediction of Blast Furnace Gas Output. Acta Automatic Sinica 35, 732–738 (2009)
5. Jaeger, H.: Reservoir Riddles: Suggestions for Echo State Network Research (Extended Abstract). In: Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, pp. 1460–1461 (2005)
6. Xue, Y., Yang, L., Haykin, S.: Decoupled echo state networks with lateral inhibition. Neural Networks 20, 365–376 (2007)
7. Ozturk, M.C., Xu, D., Principe, J.C.: Analysis and design of echo state networks. Neural Computation 19, 111–138 (2007)
8. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
9. Gray, R.T., Fung, C.K.C., Robinson, P.A.: Stability of small-world networks of neural populations. Neurocomputing 72, 1565–1574 (2009)
10. Carpenter, G.A.: Large-scale Neural Systems for Vision and Cognition. In: Proceedings of International Joint Conference on Neural Networks, USA, pp. 14–19 (2009)
11. Kaiser, M., Hilgetag, C.C.: Modelling the development of cortical systems networks. Neurocomputing 58, 297–302 (2004)
12. Center for Machine Learning and Intelligent Systems (1999), `http://archive.ics.uci.edu/ml/datasets.html`

The ART2 Network Based on Memorizing-Forgetting Mechanism

Xiaoming Ye[1,2], Xiaozhu Lin[2], and Xiaojuan Dai[3]

[1] College of Information Science and Technology,
Beijing University of Chemical Technology, Beijing, 100029
[2] College of Information Engineering,
Beijing Institute of Petrochemical Technology, Beijing, 102617
[3] National Museum of China, Beijing, 100006

Abstract. In order to imitate the course of learning and cognizing things with the human brain better, this article introduced the human brain's Memorizing-forgetting Character to the ART2 Network, then with the Memory Strength as the basis of sequence for recognition with existing patterns, thus this network model would be improved better. Through Simulation for recognition and classification with experimental samples, we prove that the ART2 network with Memorizing-forgetting Character could recognize experimental samples with less time in recognition than original ART2 network, and improve the efficiency of network.

Keywords: ART2; neural network; memorizing-forgetting mechanism; classification.

1 Introduction

ART2 is a kind of self-organizing neural network which is based on adaptive resonance theory. It carries out the classification by using competitive learning and self-steady mechanism, and can learn by itself in dynamic environment with noise and without supervision. Its learning process can recognize learned models fast and be adapted to new unknown objects rapidly[1-2]. This network is a simulation which is the course of cognizing things with the human brain[3]. In the initial perception of one thing, according to the empirical knowledge we will compare and judge it with the previous knowledge of things to ascertain the recognition of this thing; after recognizing the thing, it will be classified with certain rules[4].

As the ART2 network compared the new thing with the existed knowledge one by one, so when the number of samples is large, the time it takes will be greatly increased, thus affecting the efficiency of the network. In order to improve the efficiency of the network and simulate the process of human brain for recognizing things better, this article introduced the Memorizing-forgetting Character of human brain to the ART2 network, which improved the network model. Through the simulation of identification and classification for test samples, it could be proved that the ART2 network model which has been introduced the Memorizing-forgetting mechanism can extract the sample which appears for many times more quickly.

© Springer-Verlag Berlin Heidelberg 2011

2 The Theory of ART2 Network

2.1 Ebbinghaus Memorizing-Forgetting Curve and Its Theory

German psychologist, Hermann Ebbinghaus (1850-1909) found that the forgetting is immediately beginning after learning and the forgetting process is uneven. Initially the speed of the forgetting was very fast and then it slows down gradually. He held that "maintain and forgotten is a function of time", and drew the curve which could describe the process of forgetting according to the experimental results, that is the famous Ebbinghaus memorizing-forgetting curve. Vertical axis of the graph represents the number of knowledge that is remembered from the learning, the horizontal axis represents time (days), the curve show the regularity of the variation of memory strength[5-7].

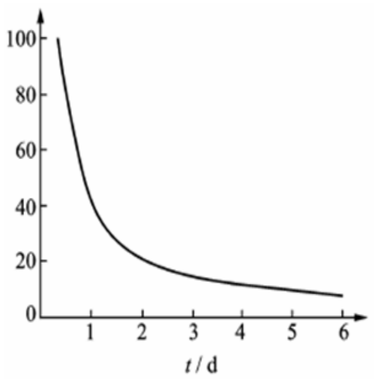


Fig. 1. Ebbinghaus memorizing-forgetting curve

This curve told that forgetting in the learning is a regular thing and the process of forgetting is uneven, not that we forget a few things one day and forget a few things another day again, but that in the initial stages of the memory the speed of forgetting is fast, then slows down gradually, and almost forget no longer after a fairly long time. The theory described above is the development rule of forgotten, that is the principle of "fast first slow second".

The initial memory strength at the time t0 is$M(t0)$, at the time t1 the memory strength attenuates to $M(t1)=M(t0)e^{-\lambda(t1-t0)}$, after being re-inspired(D), M(t1) changed as $M'(t1)$, then starting a new decay process, and

$$M'(t1) = M(t0)e^{-\lambda(t1-t0)} + D \tag{1}$$

λ is the forgetting constant, the largerλ is, the faster forgetting speed is and the shorter memory cycle is. D is the number of memory material. The PNN network which is introduced the memorizing-forgetting mechanism calculated the memory strength according to the above-mentioned principle[8].

2.2 The Working Principle of ART2 Network Which Is Introduced the Memorizing-Forgetting Mechanism and Its Description of the Algorithm

According to the principle of ART2 network, we know that the traditional ART2 network makes use of the similarity of the input sample and the existed patterns as the winning mechanism that we compare with the model of great similarity. While ART2 network is introduced the memory strength, we would make use of the memory strength of the existed patterns as winning mechanism for sorting. The input sample would be always compared with the pattern of strong memory, once identified we would not compare it with other things, then save the time of recognition. This principle is consistent with that the human brain identifies the common things quickly while it spend more time on identifying the rare things or the things that have not seen before. For this reason, the ART2 network which is introduced the memorizing-forgetting mechanism saves the time of network's recognition and improves the speed of recognition.

The input vector is i; i_i is the i'th component of the input vector; a,b,c,d,e is constant; The six sublayers of the comparison layer are w、x、v、u、p and q layer; N is the number of neurons of each sublayer of F_1 layer that is the dimensions of input; M_i is the limit of neuron patterns of F_2 layer; z_{ij} is connection weight that is from F_2 layer to p layer; z_{ji} is connection weight that is from p layer to F_2 layer; i=1,2, …,N, j =1,2, …,M_i; ρ is alert threshold. The memory strength of each pattern: Memory_j, j = 1,2,…,M_i; D is the amount of memory material; b is forgetting constant.

The parameters of ART2 network would be initialized before it is run, and the conditions for initializing of each parameter are as follows:

$a, b > 0$; $0 \le d \le 1$(Generally, $d = 0.9$) ; $cd / (1-d) \le 1$ (Generally, $c = 0.1$) ;$e << 1$;
$0<\theta<1/N^{1/2}$; $0<\rho\le 1$;

The connection weight that is from F_2 layer to p layer: $z_{ij}(0) = 0$;

The connection weight that is from p layer to F_2 layer: $z_{ji}(0) \le 1/[(1-d)*N^{1/2}]$

The calculation steps of ART2 network which is introduced the memorizing forgetting mechanism are as follows:

Step 1 Set a counter, and its initial value is 1, and all output of neurons is 0 vector;

Step 2 Input the i vector to w layer, and the output of w layer is:

$$w_i = i_i + au_i \tag{2}$$

and record the current system time at the same time;

Step 3 Transfer signal to x layer, the output of x layer is:

$$x_i = w_i / \|w\| \tag{3}$$

Step 4 Transfer signal to v layer, and the output of v layer is:

$$v_i = f(x_i) + bf(q_i) \tag{4}$$

at this moment q=0, so the second item: $bf(q_i) = 0$;

Step 5 Transfer signal to u layer, and the output of u layer is:

$$u_i = v_i / \|v\| \tag{5}$$

Step 6 Transfer signal to p layer, and the output of p layer is:

$$p_i = u_i + dz_{iJ} \tag{6}$$

J is the winning neuron of F_2 layer; if F_2 layer is not activated, $p_i = u_i$; if the network is at initial state, as $z_{iJ} = 0$, also $p_i = u_i$;

Step 7 Transfer signal to q layer, and the output of q layer is:

$$q_i = p_i / \|p\| \tag{7}$$

Step 8 Calculate the output of r layer:

$$r_i = (u_i + c\, p_i) / (\|u\| + c\|p\|) \tag{8}$$

Step 9 If $\|r\| < \rho$, send a reset signal to F_2 layer, and exclude the neuron of F_2 layer that is activated at this time, and the counter is 1, then back to step 2; if $\|r\| > \rho$and the counter is 1, the counter adds 1 and carry out step 10; if $\|r\| > \rho$and the counter is more than 1, carry out step 12, at the moment the network has arrived resonance yet;

Step 10 Calculate the output of F_2 layer:

$$\text{Memory}_j = \text{Memory}_j * e^{-\lambda(\text{time}-\text{time0})} \tag{9}$$

and calculate the max value of Memory_j. The first input sample initializes the first category of network that is the first pattern, and the memory strength: Memory_1=D, and record the memory time of this pattern: time0_J;

Step 11 Repeat step 6 to step 9;

Step 12 Modify the bottom-up weight of winning neuron of F_2 layer:

$$z_{Ji} = u_i / (1-d) \tag{10}$$

and if the input sample is new pattern, Memory_j=D; if it is old pattern, modify the memory strength according to the formula:

$$\text{Memory}_j = \text{Memory}_j * e^{-\lambda(\text{time}-\text{time0})} + D \tag{11}$$

Step 13 Modify the up-bottom weight of winning neuron of F_2 layer:

$$z_{iJ} = u_i / (1-d) \tag{12}$$

Step 14 Remove input vector, and memorize all neurons that are not occupied of F_2 layer. Back to step 1, and begin to input the new vector.

In the winning mechanism of step 10, we make use of the memory strength formula (11) in stead of the similarity formula in the traditional ART2 network

$$T_j = p_1 * z_{j1} + p_2 * z_{j2} + \ldots + p_N * z_{JN} \tag{13}$$

and record memory time of each pattern, make the program the same as the human which has a life.

3 Simulation of Experiment

3.1 The Input Samples of the Network

The samples of experiment are two-dimensional coordinates, such as (1, 1), (2, 1). The noise point is the point that is deviated from the original location, such as (1.001, 1.003), (2.002, 1.006), as Fig.2 shows.

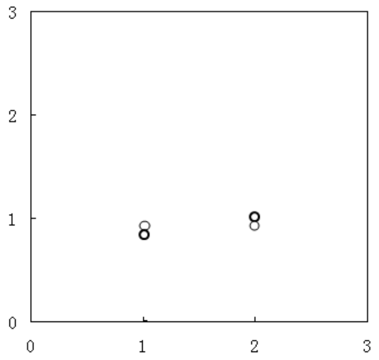


Fig. 2. Sample form of input coordinates

(1) Experimental samples 1:
(2,1),(1,2),(1,1),(2.001,1.002),(1,1),(1.007,2.004),(1.008,1.007),(1,1),(1.003,1.001),(1,2), (1,1),(1,2),(1.007,2.003),(2,1),(1,1), these are 3 patterns of (2,1),(1,2),(1,1), altogether 15 input samples.

(2) Experimental samples 2:
(1,1),(1,2),(2,1),(1,3),(3,1),(1,4),(4,1),(1,5),(5,1),(1,6),(1.001,2.002),(1,2),(1.004,1.003), (1,2),(1,1),(2,1),(1,2),(1,6),(5,1),(1,4),(3,1),(1,3),(1,2),(1,1),(1,2),(2,1),(1,3),(1,41),(1,2), (2,1),(1,3),(1,2),(3,1),(1,5),(3,1),(1,2),(2,1),(1.002,1.005),(1,2),(2.007,1.001),(1,2),(1,1), (1,2),(1,1),(1,2),(2,1),(1,2),(1,1),(1,2),(3,1),(1,2),(1,3),(1,2),(1,3),(1,2),(1,1),(1,6),(1,4), (1,2),(1,3),(3,1),(1.006,2.003),(1,1),(2,1),(1,3),(1,2),(1.007,1.008),(2,1),(1,3),(1,5),(1,4), (1,2),(1,1),(1,3),(1,1),(1,3),(1,1),(1,1),(1,2),(1,3),(1,2),(1,1),(2,1),(1,2),(1,1),(1,3), (1.004,2.001),(1,3),(1,2),(1,1),(1,6),(1,2),(1.005,6.009),(1,3),(1,6),(1.002,1.004),(1,2), (1,6),(1.003,4.001),(3,1), these are 10 patterns of (1,1),(1,2),(2,1),(1,3),(3,1),(1,4),(4,1), (1,5),(5,1),(1,6), altogether 100 input samples, and repeat these coordinates for 100 times to form 10,000 input samples.

Learn and recognize the two experimental samples above with the traditional ART2 network and the ART2 network which is introduced the memorizing-forgetting mechanism separately.

3.2 Results of Experiment and Analysis

3.2.1 Results of Experimental Samples 1 and Analysis

(1) Traditional ART2 network

Parameters: $a=10, b=10, c=0.1, d=0.9, e=0, \theta=0.1, \rho=0.9997$,

$z_{ji} = 0.08-0.001j$, $j=0,1,\ldots,M_i$.

Results of experiment: the recognition results are 3 patterns which are (2, 1), (1, 2), (1, 1), as Fig.3 shows.

(2) ART2 network which is introduced the memorizing-forgetting mechanism

Parameters: $a=10, b=10, c=0.1, d=0.9, e=0, \theta=0.1, \rho=0.9997$,

forgetting constant: $\lambda=1/60$, the amount of memory material: $D=1$.

Results of experiment: the recognition results are 3 patterns which are (2, 1), (1, 2), (1, 1), similarly as Fig.3 shows.

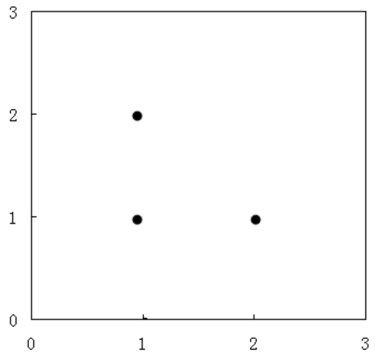


Fig. 3. Recognition patterns of experimental samples 1

After the network has recognized the tenth sample that is (1, 2), at the moment the network has learned and recognized 3 patterns which are (2, 1), (1, 2), (1, 1), and calculate the memory strength of these three patterns which are 1.9992s, 2.9984s, 4.9948s. According to the descending memory strength we sort the patterns which has been recognized, and the result is (1, 1), (1, 2), (2, 1). The pattern whose memory is strongest is the sample which appears most in ten samples. For recognizing the remaining samples, after the sample is input into the network we compare it with the pattern that is (1, 1) whose memory is strongest. If they are the same pattern, it belongs to this pattern, and we modify its weight and record the memory strength of this pattern at this moment. Then the memory strength of the pattern which matches with the input sample would be strengthened while the memory strength of other patterns would attenuate over time according to Ebbinghaus memorizing-forgetting curve. If they are not the same pattern, the input sample is compared with the pattern whose memory strength is the second strong until it is compared with all existed pattern, and now if it doesn't belong to any pattern yet, then we attribute it to a new pattern. And we record the memory strength of the new pattern as 1,while the memory strength of other patterns would also attenuate over time.

Because the number of samples is small, the experimental results only show that the ART2 network which is introduced the memorizing-forgetting mechanism can also classify correctly, but don't show the superiority of recognition time. Thus, in the experimental samples 2, we take 10,000 samples of 10 patterns as the input of the network.

3.2.2 Results of Experimental Samples 2 and Analysis

(1) Traditional ART2 network

Parameters: a=10,b=10,c=0.1,d=0.9,e=0,θ=0.1,ρ=0.9997,

$z_{ji} = 0.08-0.001j$,j=0,1,…,M_i.

Results of experiment: the recognition time is 13.406s, and the results of recognition of 10,000 samples are 10 patterns that is (1,1),(1,2),(2,1),(1,3),(3,1),(1,4),(4,1),(1,5),(5,1),(1,6), as the Fig.4 shows.

(2) ART2 network which is introduced the memorizing-forgetting mechanism

Parameters: a=10,b=10,c=0.1,d=0.9,e=0,θ=0.1,ρ=0.9997,

forgetting constant: λ=1/60, the amount of memory material: D=1.

Results of experiment: the recognition time is 9.968s. The results of recognition of 10,000 samples are 10 patterns that is (1,1),(1,2),(2,1),(1,3),(3,1),(1,4),(4,1),(1,5),(5,1),(1,6), also as the Fig.4 shows. After finishing recognition, the program records the memory strength of each pattern and sorts them according to the strong or weak memory strength. The results of sorting are as follows: (1, 2), (1, 1), (1, 3), (2, 1), (1, 6), (3, 1), (1, 4), (1, 5), (5, 1), (4,1). (1, 2) is the pattern whose memory strength is strongest that appears most in the samples, other patterns follows.

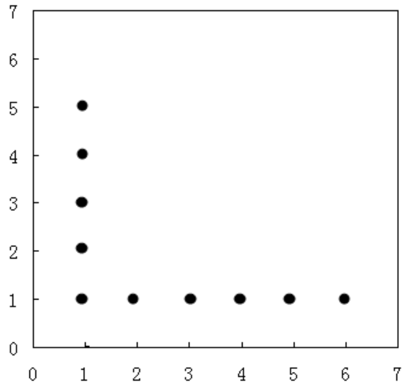


Fig. 4. Recognition patterns of experimental samples 2

From the two kinds of network's recognition results for experimental samples 2, we could see that both results of classification are right, but in the identification time the ART2 network which is introduced the memorizing-forgetting mechanism is 3.438 seconds faster in recognizing samples than traditional ART2 network. The reason is that when traditional ART2 network recognizes samples, it makes use of the similarity of the input sample and the existed patterns as the winning mechanism, and recalculates the winning neuron of the existed patterns when comparing with each pattern; while the ART2 network which is introduced the memorizing-forgetting mechanism makes use of the memory strength of the existed patterns as winning mechanism.

Therefore, when identifying a large number of samples, the ART2 network which is introduced the memorizing-forgetting mechanism improves the speed of recognition.

4 Conclusions

The results of research show that the ART2 network which is introduced the memorizing-forgetting mechanism make use of system time for calculating the memory strength of the existed pattern, and the network takes it as the winning mechanism of neuron, then it becomes a network with human characteristics which is close to the cognitive of human brain in the way of recognition. Thus, when recognizing the thing which appears frequently or the number of samples which is large, the ART2 network which is introduced the memorizing-forgetting mechanism could reduce the calculation amount, save recognition time, and improve the recognition speed of ART2 network.

References

1. Wann, C.D.: A comparative study of self-organizing clustering algorithms dig net and ART2. J. Neural Networks 10(4), 737–754 (1997)
2. Carpenter, G.A., Grossberg, S.: ART2: Self organization of stable category recognition codes for analog input patterns. J. Applied Optics. 26(23), 4919–4930 (1987)
3. Klotz, G.A., Stacey, D.A.: ART2 based classification of sparse high dimensional parameter sets for a simulation parameter selection assistant. In: 2005 IEEE International Joint Conference on Neural Networks, vol. 2, pp. 1081–1085. IEEE press, Canada (2005)
4. Altuwaijri, M.M., Bayoumi, M.A.: A thinning algorithm for Arabic characters using ART2 neural network. IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 45(2), 260–264 (1998)
5. Sikstrom, S.: Forgetting curves: Implications for connectionist models. J. Cognitive Psychology, 95–152 (2002)
6. Badiru, A.B.: Computational survey of unvaried and bipartite learning curve models. IEEE Transactions on Engineering Management 39, 176–188 (1992)
7. Jaber, M.Y., Kher, H.V.: Variant versus invariant time to total forgetting: the learn-forget curve model revisited. J. Computers & Industrial Engineering 46(4), 697–705 (2004)
8. Dai, X., Lin, X., Zhou, X.: The Pattern Neuron Network based on Memorizing-forgetting mechanism. J. Journal of Beijing Institute of Petro-Chemical Technology 18(3), 1–6 (2010)

Application of BP Neural Networks in Prediction of the Material Dynamic Properties

Xiao-ling Liu[1,2], Shun-cheng Song[2], and Ting Lei[1]

[1] School of Civil Engineering and Architecture,
Southwest University of Science and
Technology, Mianyang, Sichuan, 621010, China
liuxl-swjtu@163.com
[2] School of Mechanics and Engineering,
Southwest Jiaotong University,
Chengdu, Sichuan, 610031, China

Abstract. This paper deled with the Hopkinson bar experimental data by BP Neural Networks, including of the influence of the temperature and strain rate to the dynamic yield strength of the material. In general, the strain rate effect and the temperature effect on the material dynamic properties in the tests are coupled with each other. Through handling of the experimental data by the BP Network system, these two effects can be uncoupled and in this paper the curves of the material dynamic yield strength vs. strain rate with different temperatures and the curves of the material dynamic yield strength vs. temperature with different strain rates are presented.

Keywords: artificial neural networks; BP method; material dynamic properties.

1 Introduction

Back-propagation method (BP method) is the supervised learning algorithm that is the most widely and successfully used in feed forward network nowadays. Many researchers developed the BP method from diverse views in the different times. It has been suggested by Amari[1] that the stochastic gradient method should be introduced to train the neural network that has single hidden unit (1967). Also, the basic idea of back-propagation may be traced further back to the book *applied optimal control* by Bryson and Ho (1969)[2]. The back-propagation method is described by Werbos[3] in his Ph.D. thesis at Harvard university in August 1974.Werbo's Ph.D. thesis was the first documented description of efficient reverse-mode gradient computation that was applied to general network models with neural networks arising as a special case. From Braun's work (1994) it can be seen that Rumelhart, Hinton and Williams[4] gave first the derivation of the back-propagation, which resolved scientifically the learning and updating of the weight of multiplayer feed-forward neural network, and developed the techniques of the artificial neural networks.

© Springer-Verlag Berlin Heidelberg 2011

In this paper, the back-propagation method was used to train the data of Hopkinson bar experiments which included of the influence of the temperature and strain rate to the dynamic yield strength of the material. Through handling of the experimental data by the BP Network system, the curves of the material dynamic yield strength vs. strain rate with different temperatures and the curves of the material dynamic yield strength vs. temperature with different strain rates are presented.

2 Samples Learning and Weights Adjustment in Back Propagation Method

Back-propagation method proposed by Rumelhart and McClelland (1986) has often been used in artificial neural networks for various actual classification problems. This method is briefly described as follow.

In the application of the back propagation method, there are two distinct passes of computations. The first is referred to the forward pass, and the second is referred to the backward pass. In the forward pass, the synaptic weights remain unaltered throughout the network, and the function signals of the network are computed on a neuron-by-neuron basis. In the backward pass the error signals of the outcome layer are passed leftward through the network layer by layer. The neural network of the back-propagation consists of a number of nonlinear units in which the active function should be continuous and differentiable. The active function associated with the neurons is logistic function:

$$f(u_j) = \frac{1}{1+e^{-u_j}} = \frac{1}{1+e^{-(\sum w_i x_i - \theta_j)}} \,. \tag{1}$$

Where $f(u_j)$ is the output of the each layer, and u_j is the induced local field of neuron j and expressed by

$$u_j = \sum w_{ji} x_i - \theta_j \,. \tag{2}$$

Where w_{ji} denotes the synaptic weight connecting the outcome of neuron i to the input of neuron j and θ_j denotes the threshold value. The basis idea of back-propagation method is that the average squared value of the errors between expected responses and the function outcomes gets to a minimum by gradient search. In the BP network of three layers, let us make the assumption that there are n neurons in the input layer, n_1 neurons in the hidden layer and n_2 neurons in the outcome layer. If there are p learning samples: $x^1,\ x^2,\dots,\ x^P$ corresponding expected responses $t^1,\ t^2,\dots,\ t^P$ and the function outcome of the P1th sample is $y_l\,(l=1,2,\dots,m)$, the squared value of the error between expected response and the function outcome is:

$$E_{P_1} = \frac{1}{2}\sum_{l=1}^{m}(t_l^{p_1} - y_l^{p_1})^2 . \tag{3}$$

The squared value of the total error of the p learning samples is:

$$E_{总} = \frac{1}{2}\sum_{p=1}^{p}\sum_{l=1}^{m}(t_l^{p_1} - y_l^{p_1})^2 . \tag{4}$$

The weight correction is defined by the delta rule:

$$\Delta w_{ji} = -\sum_{p}\eta\frac{\partial E_T}{\partial w_{ji}} . \tag{5}$$

To increase the learning efficiency of the back propagation method and avoid the danger of instability, a simple method that can modify the delta rule by adding a momentum term is proposed. The expression is:

$$\Delta w_{ji}(n) = \alpha\Delta w_{ji}(n-1) + \eta\delta_j(n)y_j(n) . \tag{6}$$

So the synaptic weights of the network in the layer l are:

$$w_{ji}^{(i)}(n+1) = w_{ji}^{(l)}(n) + \alpha[\Delta w_{ji}^{(l)}(n-1)] + \eta\delta_j^{(l)}(n)y_j^{(l-1)}(n) . \tag{7}$$

Because the adjustment of synaptic weights are carried out in terms of the total error between expected response and the function outcome after learning all of samples input to the network, this modification is called batch mode. This modification may ensure E_T decreasing. So the convergence is faster one by one when the learning samples are more.

From the above analysis, we can see that the learning process of the neural network includes two steps: the first step is computation of the function outcome of each layer neurons from input layer. The second step is to modify the synaptic weights from outcome layer in terms of the computational error signals.

3 Program Performed

The key of using the BP method is processing of the samples learning and weights adjustment. The Program performed is following.

- Initialization. The synaptic weights and threshold values from a smaller uniform distribution are chosen as initial values.
- Presentations of training samples. The training samples are input into the network and the function signal of the each layer neurons is computed.
- Computation of the errors of the outcome layer.

- Computation of the total error. The total error is computed after all of the training samples input into the network. If the total error satisfies the need of the precision, training stops. If not, the following step is performed.
- Backward computation. The outcome error of the each layer in the network is computed as follow.

$$\delta_2 = (t_l^{p_1} - p_l^{p_1}) y_l^{p_1} (1 - y_l^{p_1}) . \tag{8}$$

- Where δ_2 is the error of the neuron j in The outcome layer l.

$$\Delta w_{ji} = -\sum_p \eta \frac{\partial E_T}{\partial w_{ji}} . \tag{9}$$

Where δ_1 is the error of neuron j In the implicit layer l.
- Re-computations. The synaptic weights and threshold values are adjusted and the computations return to the step 3.

Fig. 1 shows the computational processes.

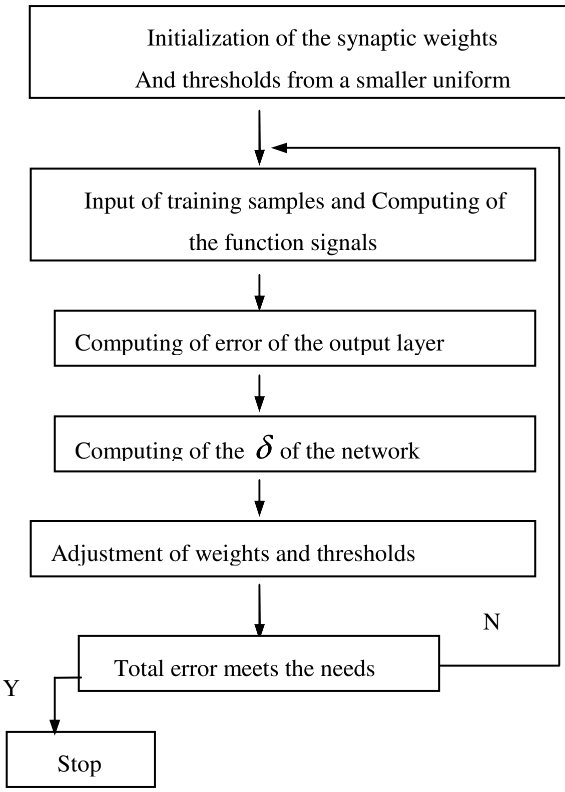

Initialization of the synaptic weights
And thresholds from a smaller uniform
Input of training samples and Computing of the function signals
Computing of error of the output layer
Computing of the δ of the network
Adjustment of weights and thresholds
Total error meets the needs
N
Y
Stop

Fig. 1. Flow diagram of the BP networks

4 Prediction of Material Dynamic Properties

Material dynamic properties with strain rate less than 104/s are able to tested by Hopkinson bar. Recently we have measured the dynamic properties of a armor steel with the apparatus. The dynamic yield strengths of the steel under different temperature (*T*) and with different strain rate ($\dot{\varepsilon}$) are measured and shown in table 1, where σ_y^d / σ_y^s denotes the ratio of dynamic strength to static strength. In general the temperature and strain rate in the Hopkinson bar experiments are not easy to be controlled accurately and so it is difficult to get the separate values with various strain rates but under the identical temperatures except under the room temperature, and especially to get the values under various temperatures but with the identical strain rates. Namely, it is difficult to uncouple the temperature effect and the strain rate effect on the dynamic yield strength of the steel.

The dynamic yield strength of the armor steel under different temperatures and with different strain rates could be predicted in terms of the known data of the Hopkinson bar experiments, and the temperature effect and strain rate effect could be uncouple with the BP method network. To compute simply, the temperatures in table 1 are normalized by 1800K(the melt temperature of the steel) and the strain rates are normalized by 10^4/s and the dynamic yield strength are normalized by 860Mpa(static yield strength). The normalized data are shown in table 2.

After the data are normalized, they could be input into the BP network. In this paper, there are 3 layers in the BP network where the input layer has 2 neurons in which the normalized temperatures and strain rates are input and the outcome layer has 1 neuron in which the normalized dynamic yield strengths are input. It is noted that the outcome values are standard but the final results are converted back into the values with dimensions in the program for treatment of the data straight-forwards.

Table 1. Experimental data of a armor steel

Test Numbers	*T/K*	$\dot{\varepsilon}$ /(*1/S*)	σ_y^d / σ_y^s
1	223	1000	1.29
2	243	900	1.24
3	300	800	1.17
4	400	1200	1.16
5	408	940	1.14
6	600	1100	1.0
7	609	1080	0.98
8	300	1050	1.22
9	300	2000	1.26
10	300	3200	1.31
11	800	1100	0.83
12	900	1100	0.74

Table 2. Learning samples (Normalized experimental data)

Sample Numbers	Normalized Temperatures	Normalized Strain rates	Normalized Yield Strengths	
			Expected outcomes	Actual outcomes
1	0.124	0.100	0.898955	0.876601
2	0.135	0.090	0.864111	0.862239
3	0.167	0.080	0.815331	0.836824
4	0.222	0.120	0.808362	0.812917
5	0.227	0.094	0.794425	0.800582
6	0.333	0.110	0.696864	0.704921
7	0.372	0.108	0.682927	0.661911
8	0.167	0.105	0.850174	0.847003
9	0.167	0.20	0.878049	0.891433
10	0.167	0.320	0.912892	0.914339
11	0.444	0.110	0.578397	0.582101
12	0.500	0.110	0.515679	0.518277

Table 3. Test results of computations with the BP network

Normalized Temperatures	Normalized Strain rates	Normalized Yield Strengths		Errors / %
		Predicted Values	Experimental values	
0.167	0.2008	0.872012	0.881494	1.087
0.167	0.2565	0.895528	0.898188	0.297
0.167	0.3925	0.938345	0.929376	0.956
0.372	0.1080	0.670863	0.661911	1.334

The 12 samples in the table 2 are input into the BP network and the learning error can be reach of 0.000888 by iteration of 200001 steps with step length 0.05, and momentum term 0.8.

The 4 samples were taken to test. The computational values and the experimental values in the samples are shown in table 3. From the table 3, it can be seen that the max error is 1.334% that has reached the required precision.

The dynamic yield strengths of the armor steel under various temperatures were computed with the strain rates 500/s, 2000/s, 3000/s respectively and 3 curves of relationship of dynamic yield strength to temperature are plotted in Fig. 2. Fig. 3 shows the relationship of the dynamic yield strength vs. the strain rate for the steel under the temperatures 223K, 273K, 573K respectively, where the strain rates are taken up with the values of logarithm.

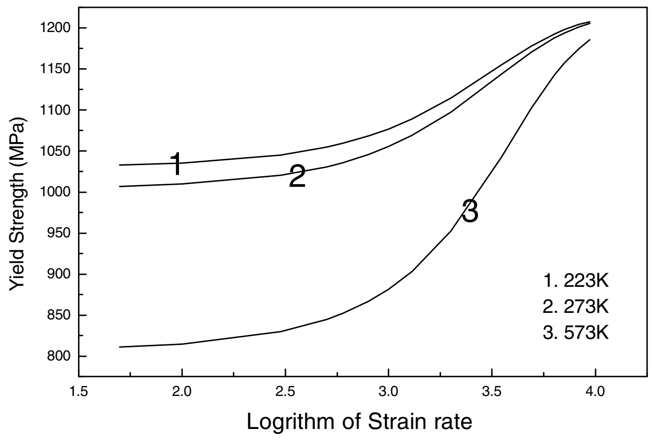


Fig. 2. Plots of yield strength vs. temperature

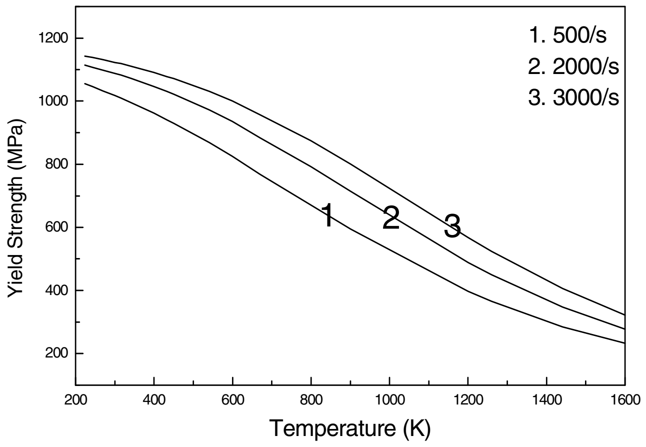


Fig. 3. Plots of yield strength vs. strain rate

5 Conclusion

From the prediction results of the armor steel, it could be seen that the effects of the temperature and strain rate on the dynamic yield strength of the armor steel are nonlinear. As strain rate unchanging, the dynamic yield strength is decreasing with temperature increasing. When the temperature is nearby the melt temperature of the steel, its yield strength is approximate to zero. In Fig. 2 this trend is shown. In Fig. 3 the strain rate effects are presented. It is obvious that as temperature unchanging, the dynamic yield strength is increasing with the strain rate increasing.

It could be seen from this work that the nonlinear relationship of some quantities could be predicted by the BP method. So the BP network may be a useful tool to resolve some complex, fuzzy and stochastic problems.

References

1. Amari, S.I.: A theory of adaptive pattern classification. IEEE Trans. on Electronic Computers EC-16, 299–307 (1967)
2. Bryson, A., Ho, Y.C.: Applied optimal control, New york (1969)
3. Werbos, P.: New tools for prediction and analysis in the behavioral science [dissertation], Harvard University (1974)
4. Braun, H.: ENZO-M, A hybrid approach for optimizing neural networks by evolution and learning. In: Davidor, Y., Männer, R., Schwefel, H.-P., et al. (eds.) PPSN 1994. LNCS, vol. 866, pp. 440–451. Springer, Heidelberg (1994)

Spaces Enhance Word Segmentation and Comprehension in Tacit Reading

Mayumi Toshima[1], Tetsuo Ishikawa[2,3,4], and Ken Mogi[4]

[1] Department of Informatics,
Graduate University for Advanced Studies
2-1-2 Hitotusbashi, Chiyoda-ku, Tokyo, 101-8430, Japan
mamitako@nii.ac.jp
[2] Department of Computational Intelligence and Systems Science,
Tokyo Institute of Technology
G3-46, 4259 Nagatsuta-cho, Midori-ku, Yokohama, 226-8502, Japan
i@brn.dis.titech.ac.jp
[3] Centre for Advanced Research on Logic and Sensibility, Keio University
2nd Floor Mita Toho Bldg. 3-1-7 Mita, Minato-ku, Tokyo, 108-0073, Japan
[4] Sony Computer Science Laboratories, Inc.
Takanawa Muse Bldg. 3-14-13 Shinagawa-ku, Tokyo, 141-0022, Japan
kenmogi@csl.sony.co.jp

Abstract. There are a lot of types of written language systems all over the world. The Japanese writing system uses ideographic and phonetic symbols without spaces between words, where the ability of finding the boundary between words is expected to have a positive correlation with the ability of general language comprehension. Is such a correlation to be found in languages (e.g., English) where words are conventionally separated by spaces between them? Here we report a study on how spaces put on the boundary of two adjacent words enhance tacit reading texts. We conducted a "slash task" where our subjects whose first language was Japanese were instructed to place a slash between adjacent words of sentences written continuously without spaces between the words in typical, short and easy sentences. Our data illuminate the role of "active segmentation" in language comprehension.

Keywords: Language, word, segmentation, learning, tacit reading.

1 Introduction

Languages are important elements in human cognition, and play an essential role in the communication among humans. In the process of cognitive development of one's mother tongue, a progress in the ability of the first language is accompanied by an increasing difficulty in acquiring other languages. Such a cognitive phenomenon occurs because of the characteristics of our brains [1][2][3]. The nature of the first language acquired results in significant differences in the construction of efficient neural network systems, which in turn affect second language acquisition [4][5][6].

© Springer-Verlag Berlin Heidelberg 2011

Reading words consist of perceiving language symbols visually and assigning them appropriate meanings. Some of the difficulties encountered in second language acquisition are due to the differences in the system of writing between the languages. Among the various languages in the world, there are two distinctive systems of writing, i.e., ideographs and phonograms. In both of these writing systems, sentences are constructed of strings of words. In Japanese writing, the segmentation between the words is given implicitly, where the letters are written continuously without placing spaces between the words; both of the Chinese letters [7] which contain ideograms and phonograms and the Thai letters [8] which contain phonograms do not need spaces as well. In English and other European languages, spaces are placed between the words. It is interesting to compare the cognitive process of language segmentation between these two types of writing systems, i.e., one with spaces and one without spaces between words. Previous studies indicate that unspaced English texts influenced the speed of oral reading [8] and eye movement behaviors [9].

In the domain of speech perception, continuous auditory sequences contain no "spaces" (i.e., pauses). However it is known that (i) 8-months old infants can recognize the word boundaries [10], (ii) ERP N400 is correlated with word onset in continuous auditory stream [11] and (iii) the recursive neural network can find English word boundaries in unspaced texts [12].

Here we report a study on effects of spaces enhancing segmentation in tacit reading. In order to conduct the experiments, we invented "slash tasks"; when one finds word boundary in unspaced text, he or she is required to put a slash down on the boundary. To measure one's reading competency, several types of tests have been conducted. Most of them (e.g., [8]) require reading strings of letters aloud. Therefore, those tests depend not only on reading competence but also on speaking ability. Our slash tasks, on the other hand, only require the subjects to mark slashes between words while reading tacitly. Subjects were university students whose first language was Japanese. The subjects conducted a "slash task" where they were instructed to place a slash between adjacent words of typical English sentences written continuously without spaces between the words.

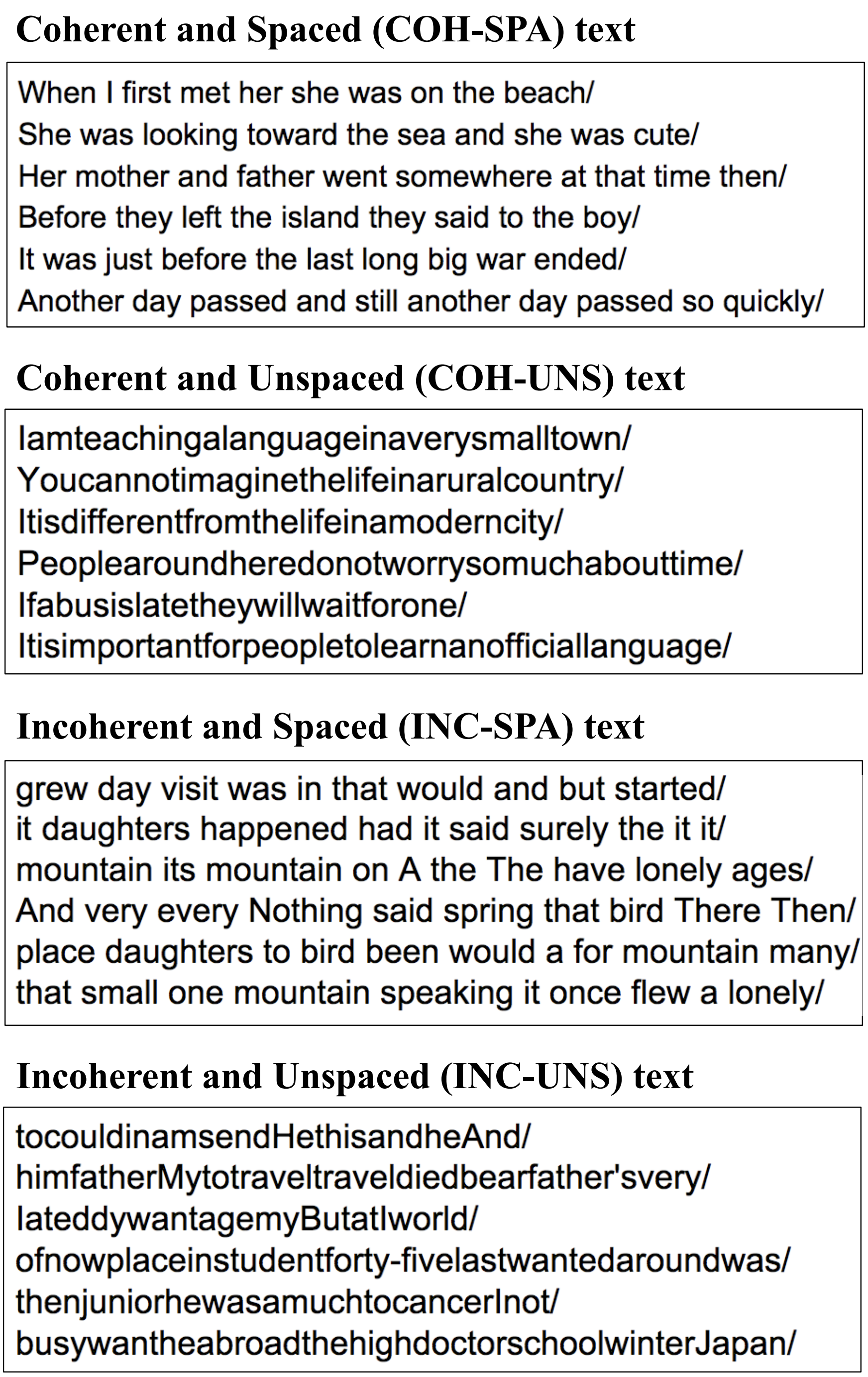
2 Experiment

Materials and Methods

32 subjects (12 females, mean age 19.6±1.7) whose native language was Japanese participated. All were undergraduate students, right-handed by self report, with normal or corrected to normal vision.

Two independent variables were designed in the experiments. (1) Spacing: spaced or unspaced texts, all written in English; (2) Coherence: coherent or incoherent texts (Fig. 1). The order of tasks was counterbalanced with coherence.

The subjects were asked to put slashes between words in a coherent/incoherent text with/without spaces within four minutes. Reading time was recorded by self-report. After that, all of the subjects took a word recognition test, making an



Coherent and Spaced (COH-SPA) text

When I first met her she was on the beach/
She was looking toward the sea and she was cute/
Her mother and father went somewhere at that time then/
Before they left the island they said to the boy/
It was just before the last long big war ended/
Another day passed and still another day passed so quickly/

Coherent and Unspaced (COH-UNS) text

Iamteachingalanguageinaverysmalltown/
Youcannotimaginethelifeinaruralcountry/
Itisdifferentfromthelifeinamoderncity/
Peoplearoundheredonotworrysomuchabouttime/
Ifabusislatetheywillwaitforone/
Itisimportantforpeopletolearnanofficiallanguage/

Incoherent and Spaced (INC-SPA) text

grew day visit was in that would and but started/
it daughters happened had it said surely the it it/
mountain its mountain on A the The have lonely ages/
And very every Nothing said spring that bird There Then/
place daughters to bird been would a for mountain many/
that small one mountain speaking it once flew a lonely/

Incoherent and Unspaced (INC-UNS) text

tocouldinamsendHethisandheAnd/
himfatherMytotraveltraveldiedbearfather'svery/
lateddywantagemyButatIworld/
ofnowplaceinstudentforty-fivelastwantedaroundwas/
thenjuniorhewasamuchtocancerInot/
busywantheabroadthehighdoctorschoolwinterJapan/

Fig. 1. Stimuli (4 types of texts)

"old" or "new" judgment. Random sampled words were presented in this test and the subjects were required to judge whether the words had been shown in the previous slash task text ("old") or not ("new"). Four sessions were conducted without intervals. The same subjects were required to take another type of test in twenty minutes; this contained thirty questions taken from a TOEIC problem book. TOEIC (Test of English for International Communication) is a widely used test of English to measure learners' ability to use English as a language system (grammar, vocabulary and usages).

The data were examined using ANOVAs with Space and Coherence as factors. Then, Games-Howell post-hot multiple comparisons were conducted. The significance level was set at 0.05.

3 Results

We found that the main effects of Coherence ($p = 0.031$) and Space ($p < 0.001$) show significances on the reading time, but their interaction indicates no significance ($p = 0.087$). The mean reading time of COH-SPA (the Coherent-Spaced condition) and INC-SPA (the Incoherent-Spaced condition) were not significantly different ($p = 0.73$). Those two conditions required shorter mean time than COH-UNS (the Coherent-Unspaced condition) ($p < 0.001$) and INC-UNS (the Incoherent-Unspaced condition) ($p < 0.001$). The mean reading time of COH-UNS and INC-UNS were rather comparable ($p = 0.28$) (Fig. 2).

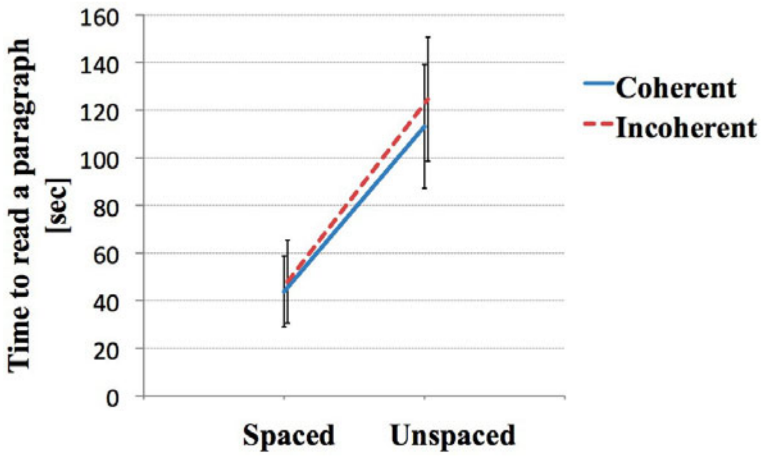


Fig. 2. Average reading time of 4 types of the texts (Mean±SD)

The number of errors in reading was the number of failures to find word boundaries. The main effects of Coherence ($p = 0.007$) and Space ($p < 0.001$) on the number of errors and also their interaction ($p < 0.001$) were found. According to the multiple comparisons, the amounts of the errors indicated significant

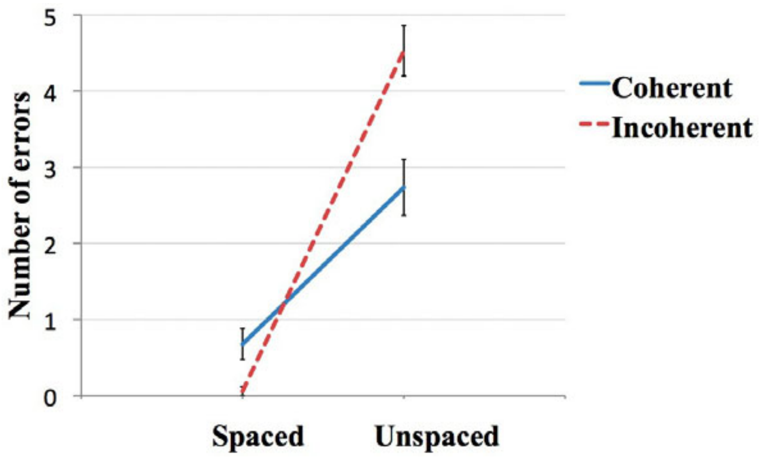


Fig. 3. Errors in reading of 4 types of the texts (Mean±SEM)

differences among all of the combinations of each condition: INC-SPA < COH-SPA < COH-UNS < INC-UNS (INC-SPA vs. COH-SPA: p = 0.03, COH-SPA vs. COH-UNS: p < 0.001, COH-UNS vs. INC-UNS: p < 0.001) (Fig. 3).

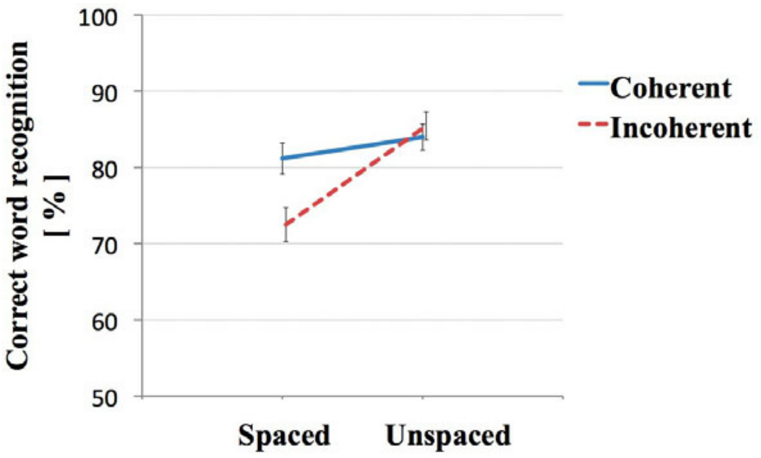


Fig. 4. "Old" or "new" judgment performances (Mean±SEM)

The correct rate of word recognition was defined as the rate of correct-old ("hit") and correct-new ("correct rejection") responses. Note that the chance level of word recognition accuracy was 50 percent because of the two-alternative ("old" or "new") forced choice. The main effect of Space on the correct rate of word recognition was significant (p < 0.001), but that of Coherence was not significant (p = 0.078). The interaction between them was significant (p = 0.003).

The multiple comparisons showed that the INC-SPA correct rate of word recognition, the old and new judgment performance was lower than other conditions (INC-SPA vs. COH-SPA: $p = 0.03$, INC-SPA vs. COH-UNS: $p = 0.001$, INC-SPA vs. INC-UNS: $p < 0.001$) (Fig. 4).

4 Discussion

Spaced texts and coherent texts were read faster than unspaced texts and incoherent texts, respectively. It is suggested that spaces work efficiently as a sort of punctuation systems. Without spaces, the subjects had to find the blocks of letters with meanings; it was necessary for the subjects to segment the words attentively and intentionally.

When doing the tasks without spaces, most of the subjects were observed leaned toward the texts and putting the tip of the pen closer to the written letters, as if they were tracing invisible lines drawn under the strings of letters. These behaviors were not observed when they were doing the tasks with spaces. Without spaces, the subjects had to be more careful to find the segments of the letters with meanings; therefore, their postures were different from those of when doing the tasks with spaces.

For the all subjects, unspaced English sentences are rather unfamiliar even if they see Japanese (their first language) sentences which have no spaces between characters and letters; they are sufficiently used to reading Japanese with its ideographic and phonetic symbols. Their scores of reading "Hiragana" and "Katakana" (phonetic symbols) and "Kanji" (ideographic symbols) are not significantly varied. The relationship between the first language and the second language word segmentation competence should be clarified in the further researches.

Errors were observed in all of the conditions. No subjects got perfect scores in all tasks. The least errors were observed when doing "Incoherent and Spaced (INC-SPA)" task, not "Coherent and Spaced (COH-SPA)" one. This may suggest that the subjects should pay more attention when they are given more unfamiliar or unusual texts than familiar and usual texts.

The correlation between "the speed and correctness" and "TOEIC scores" showed no significance in the present study. However, according to preceding researches (data not shown), it is possible to think that the ability of recognizing the blocks of words that we call it "active segmentation" should be correlated with certain cognitive and comprehension system when reading written languages.

The slash task has a possibility of application to language education and clinical assessment of aphasia and dyslexia [13][14] as it is not explicitly influenced by articulating or speaking ability. It also has the following advantages: (i) Only a pen and a bundle of paper are needed, and no complicated and expensive devices such as eye movement trackers are necessary. (ii) It requires at most about 120 seconds. (iii) It is possible to conduct classroom experiments with many participants simultaneously.

Acknowledgments. We thank Noriko Kando, Tomomitsu Herai, Eiichi Hoshino, Shinichi Nozawa, Ayako Onzo, Kei Omata, Takayasu Sekine, Tamami Sudo, Tsugumi Takano, Fumiko Tanabe, Fumihiko Taya, Toru Yanagawa for their helpful discussion.

References

1. Deacon, T.W.: The symbolic species: the co-evolution of language and the brain. W.W. Norton & Co., New York (1997)
2. Crinion, J., Turner, R., Grogan, A., Hanakawa, T., Noppeney, U., Devlin, J.T., Aso, T., Urayama, S., Fukuyama, H., Stockton, K., Usui, K., Green, D.W., Price, C.J.: Language control in the bilingual brain. Science 312, 1537–1540 (2006)
3. Osterhout, L., Poliakov, A., Inoue, K., McLaughlin, J., Valentine, G., Pitkanen, I., Frenck-Mestre, C., Hirschensohn, J.: Second-language learning and changes in the brain. Journal of Neurolinguistics 21(6), 509–521 (2008)
4. Cummins, J.: The cross-lingual dimensions of language proficiency: Implications for bilingual education and the optimal age issue. TESOL Quarterly 4, 75–87 (1980)
5. Cummins, J.: Bilingualism and minority language children. Ontario Institute for Studies in Education, Toronto (1981)
6. Nakada, T., Fujii, Y., Kwee, I.L.: Brain strategies for reading in the second language are determined by the first language. Neuroscience Research 40(4), 351–358 (2001)
7. Bai, X., Yan, G., Liversedge, S.P., Zang, C., Rayner, K.: Reading spaced and unspaced Chinese text: evidence from eye movements. Journal of Experimental Psychology: Human Perception and Performance 34(5), 1277–1287 (2008)
8. Kohsom, C., Gobet, F.: Adding spaces to Thai and English: Effects on reading. Proceedings of the Cognitive Science Society 19, 388–393 (1997)
9. Rayner, K., Fischer, M.H., Pollatsek, A.: Unspaced text interferes with both word identification and eye movement control. Vision Research 38(8), 1129–1144 (1998)
10. Saffran, J.R., Aslin, R.N., Newport, E.L.: Statistical Learning by 8-Month-Old Infants. Science 274, 1926–1928 (1996)
11. Abla, D., Katahira, K., Okanoya, K.: On-line Assessment of Statistical Learning by Event-related Potentials. Journal of Cognitive Neuroscience 20, 952–964 (2008)
12. Elman, J.L.: Distributed Representations, Simple Recurrent Networks, And Grammatical Structure. Machine Learning 7(2), 195–225 (1991)
13. Eden, G.: Presentation at Fluency Conference. Dyslexia Research Foundation, Crete (2000)
14. Wolf, M.: Proust and the Squid – the story and science of the reading brain. Harper, New York (2007)

Intelligent Semantic-Based System for Corpus Analysis through Hybrid Probabilistic Neural Networks

Keith Douglas Stuart[1], Maciej Majewski[2], and Ana Botella Trelis[1]

[1] Polytechnic University of Valencia, Department of Applied Linguistics
Camino de Vera, s/n, 46022 Valencia, Spain
{kstuart,apbotell}@idm.upv.es

[2] Koszalin University of Technology, Faculty of Mechanical Engineering
Raclawicka 15-17, 75-620 Koszalin, Poland
maciej.majewski@tu.koszalin.pl

Abstract. The paper describes the application of hybrid probabilistic neural networks for corpus analysis which consists of intelligent semantic-based methods of analysis and recognition of word clusters and their meaning. The task of analyzing a corpus of academic articles was resolved with hybrid probabilistic neural networks and developed word clusters. The created prototypes of word clusters provide the probabilistic neural networks with possibilities of recognizing corpus clusters. The established corpus comprises 1376 articles, from specialist leading SCI-indexed journals, and provides representative samples of the language of science and technology. In this paper, a review of selected issues is carried out with regards to computational approaches to language modelling as well as semantic patterns of language. The paper features semantic-based recognition algorithms of word clusters of similar meanings but different lexico-grammatical patterns from the established corpus using multilayer neural networks. The paper also presents experimental results of word cluster semantic-based recognition in the context of phrase meaning analysis.

Keywords: corpus analysis, artificial intelligence, probabilistic neural networks, semantic networks, phrase meaning analysis, natural language processing, applied computational linguistics.

1 Introduction

The hypothesis that modelling a language involves probabilistic representations of allowable sequences, determines two areas of knowledge that might be applied to text analysis. One is word clusters: it is often the case that strings of words are repeated or tend to cluster together for semantic and/or syntactic reasons. The other is the fact that given a sequence of words one might want to try and predict the next word based on what restrictions exist on the choice of next word. Another way of putting this might be that given a sequence of possible words,

© Springer-Verlag Berlin Heidelberg 2011

estimate the probability of that sequence. In a corpus of size N, the assumption is that any combination of n words is a potential n-gram. Each n-gram in our model is a parameter used to estimate probability of the next possible word. Low frequency n-grams are the most frequent. In other words, it is very common to find strings that have low frequency. In the same way, it is very common to find words that only occur once in a corpus (hapax legomena).

A corpus is a collection of linguistic data, consisting of a large and structured set of texts, which can be used for linguistic description or as a means of verifying hypotheses about a language [7,10,12]. Text corpus is nowadays usually electronically stored, distributed and processed for statistical analysis and used for checking occurrences or validating linguistic rules on a specific universe [16]. We have been doing research into word clusters in a corpus of 1376 academic articles and we have found that repetition is constant across many word sequences.

Our corpus comprises 1,376 articles, from specialist leading journals (a total of 6,104,323 tokens, 71,516 types, and 1.17 type/token ratio). The articles have all been published in journals cited in the Science Citation Index (SCI). They have been distributed in 23 knowledge areas, each of which constitutes per se a sub-corpus. They are representative samples of the language of science and technology. The corpus has been tagged with meta-textual information and transferred to an Access database by means of an application in Visual Basic.

Once the corpus had been designed and implemented, we proceeded to analyse the data by creating wordlists of technical and semi-technical terms through frequency counts and keyword identification. This process involved initially comparing a general English wordlist (from the 100 million BNC corpus) with a wordlist from our corpus. Frequencies were compared and a keyword list was created from our corpus. Analysis was conducted by processing both the corpus as a whole and each of the subject areas separately. Then, we proceeded to generate 3 to 8 word clusters (n-grams), which were transferred to a database specially designed to carry out queries related to the clusters. Furthermore, we carried out research into collocational structures which are obtained by calculating the total number of times a word is found in the neighbourhood of the node word using as the default collocation horizon 5 words to the left and 5 words to the right of the node word (although it is possible to calculate collocations using much larger horizons). Both clusters and collocational structures provide clues to lexico-grammatical patterns. For this paper, we have mainly used the data from the 3 to 8 word clusters.

The aim of the research is intelligent corpus analysis through meaning recognition of word clusters using artificial intelligence methods. We have developed a method which allows for development of possible word cluster components in a corpus for training hybrid probabilistic neural networks. The networks are capable of recognizing word clusters with similar meaning but different lexico-grammatical patterns. In other words, we are working with the idea that there is a strong tendency for sense and syntax to be associated [17]. Corpus

Linguistics needs computational tools to be able to map the close association between pattern and meaning and neural networks are ideal for pattern recognition and, consequently, semantic meaning.

2 The State of the Art

Automated tools are used by researchers for analyzing corpus frequency data. They have analyzed the differences between various registers of corpora such as fiction and academic writing and have found that many features of corpora differ between registers [1,2,3]. The features they discuss range from syntactic, through lexical, to discourse.

Parsed corpora have made it possible to generate more reliable syntactic frequency information. Much of the work with that data has looked at the frequencies of specific structures occurring with specific verbs [6,10].

Previous work on corpus analysis faced several limitations: the number of words covered, the number of structures covered, and limits on the amount of data available for low frequency items imposed by the size of the corpora [5,7,12]. While some work [11,15] has used data from larger corpora, it is an important goal to develop new reliable and efficient automatic extraction methods. Towards this goal, various automated tools have been developed during the last few years. However, most of them use old-fashioned methods, lacking functionalities such as sophisticated capabilities which could be delivered with use of artificial intelligence methods [8,9,22,23,24,25,26]. This paper proposes an approach to deal with the above mentioned problem.

3 Description of the Method

The proposed intelligent semantic-based system for corpus analysis shown in abbreviated form on Fig. 1a, consists of two subsystems: statistical corpus processing and intelligent corpus processing [26].

In the corpus processing subsystem, words are isolated from text extracted from the corpus, which are developed into various combinations of word clusters based on the statistical models of word sequences. The developed word clusters representing appropriate N-gram models are processed further for training hybrid probabilistic neural networks with learning patterns of words and clusters.

In the intelligent corpus processing subsystem, text is retrieved from the corpus using a parser. In the next step, word clusters are extracted by the parser using lexical and grammar patterns. The separated words are processed for letter strings isolated in segments as possible cluster word components. This analysis has been carried out using Hamming neural networks. The output data of the analysis consists of processed word segments. Individual word segments treated here as isolated possible components of the cluster words are inputs (Fig. 1b) of hybrid probabilistic neural networks for recognizing words. The networks use learning files containing words and are trained to recognize words as word cluster components, with words represented by output neurons.

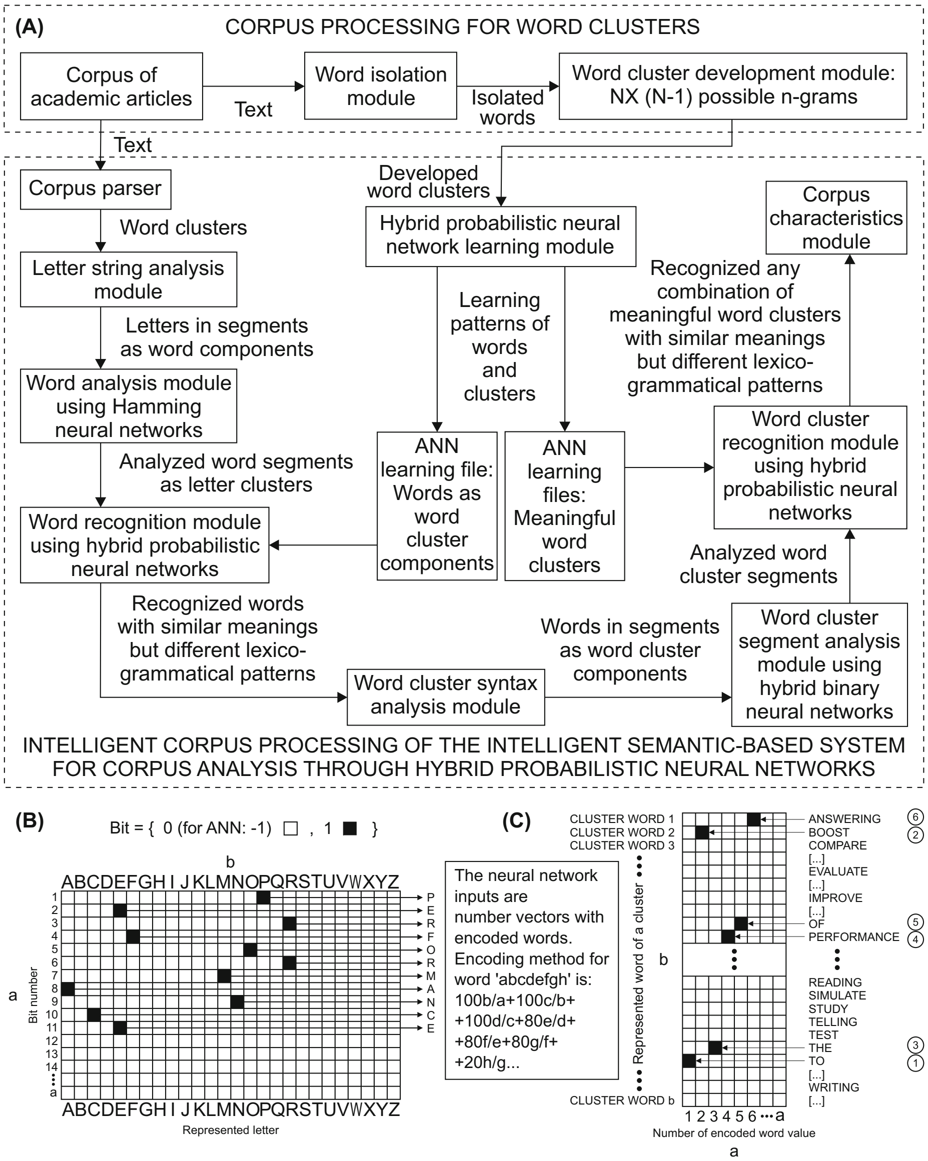

(A)
CORPUS PROCESSING FOR WORD CLUSTERS
Corpus of academic articles
Text
Word isolation module
Isolated words
Word cluster development module: NX (N-1) possible n-grams
Text
Corpus parser
Word clusters
Letter string analysis module
Letters in segments as word components
Word analysis module using Hamming neural networks
Analyzed word segments as letter clusters
Word recognition module using hybrid probabilistic neural networks
Recognized words with similar meanings but different lexico-grammatical patterns
Word cluster syntax analysis module
Words in segments as word cluster components
Word cluster segment analysis module using hybrid binary neural networks
Analyzed word cluster segments
Word cluster recognition module using hybrid probabilistic neural networks
Recognized any combination of meaningful word clusters with similar meanings but different lexico-grammatical patterns
Corpus characteristics module
Developed word clusters
Hybrid probabilistic neural network learning module
Learning patterns of words and clusters
ANN learning file: Words as word cluster components
ANN learning files: Meaningful word clusters
INTELLIGENT CORPUS PROCESSING OF THE INTELLIGENT SEMANTIC-BASED SYSTEM FOR CORPUS ANALYSIS THROUGH HYBRID PROBABILISTIC NEURAL NETWORKS
(B)
Bit = { 0 (for ANN: -1) □ , 1 ■ }
b
ABCDEFGHIJKLMNOPQRSTUVWXYZ
a
Bit number
P E R F O R M A N C E
Represented letter
(C)
CLUSTER WORD 1
CLUSTER WORD 2
CLUSTER WORD 3
CLUSTER WORD b
Represented word of a cluster
The neural network inputs are number vectors with encoded words. Encoding method for word 'abcdefgh' is: 100b/a+100c/b+ +100d/c+80e/d+ +80f/e+80g/f+ +20h/g...
b
ANSWERING 6
BOOST 2
COMPARE
[...]
EVALUATE
[...]
IMPROVE
[...]
OF 5
PERFORMANCE 4
READING
SIMULATE
STUDY
TELLING
TEST
THE 3
TO 1
[...]
WRITING
[...]
1 2 3 4 5 6 ··· a
Number of encoded word value
a

Fig. 1. (A) Diagram of the proposed intelligent semantic-based system for corpus analysis, (B) inputs of the word recognition module, (C) inputs of the word cluster recognition module

The intelligent cluster word recognition method allows for recognition of words with similar meanings but different lexico-grammatical patterns. In the next stage, the words are transferred to the word cluster syntax analysis module. The module creates words in segments as word cluster components properly, which are coded as vectors. Then they are processed by the module for word cluster segment analysis using hybrid binary neural networks. The analyzed word cluster segments become inputs of the word cluster recognition module using hybrid probabilistic neural networks (Fig. 1c). The module uses multilayer probabilistic neural networks, either to recognize the cluster and find its meaning or else it fails to recognize it. The neural networks of this module use learning files containing patterns of possible meaningful word clusters. The intelligent analysis and processing allow for recognition of any combination of meaningful word clusters with similar meanings but different lexico-grammatical patterns. The overall detailed results of the intelligent analysis are subject to processing for corpus characteristics and its linguistic description including: statistical analysis, checking occurrences, and validating linguistic rules.

The proposed intelligent semantic-based system for corpus analysis contains hybrid probabilistic neural networks which are pattern classifiers. They can become effective tools for solving classification problems of lexico-grammatical structures in corpus linguistics, where the objective is to assign cases of clusters of letters or words to one of a number of discrete cluster classes. Pattern classifiers place each observed vector of cluster data x in one of the predefined cluster classes k_i, i=1, 2, ..., K where K is the number of possible classes in which x can belong. The effectiveness of the cluster classifier is limited by the number of data elements that vector x can have and the number of possible cluster classes K. The Bayes pattern classifier implements the Bayes conditional probability rule that the probability $P(k_i\,|x\,)$ of x being in class k_i is given by (1):

$$P(k_i\,|x\,) = \frac{P\,(x\,|k_i\,)\;P\,(k_i)}{\sum_{j=1}^{K} P\,(x\,|k_j\,)\;P\,(k_j)} \tag{1}$$

where $P(x\,|k_i\,)$ is the conditioned probability density function of x given set k_i, $P(k_j)$ is the probability of drawing data from class k_j. Vector x is said to belong to a particular class k_i if $P(k_i\,|x\,) > P(k_j\,|x\,)$, $\forall j = 1,\ 2,\ \ldots,\ K,\ j \neq i$. This classifier assumes that the probability density function of the population from which the data was drawn is known a priori. This assumption is one of the major limitations of implementing Bayes classifier.

The probabilistic neural network was first introduced by Specht [18, 19, 20, 21], who was inspired by the work of Parzen [14]. The network is interesting, because it is possible to implement and develop numerous enhancements, extensions, and generalizations of the original model. It offers a way to interpret the network's structure in the form of a probabilistic density function. The probabilistic neural network simplifies the Bayes classification procedure by using a training set of clusters for which the desired statistical information for implementing Bayes classifier can be drawn. The desired probability density function of the cluster class is approximated by using the Parzen windows approach

[4,13,14]. The probabilistic neural network learns to approximate the probability density function of the cluster training samples. It should be interpreted as a function that approximates the probability density of the underlying cluster samples distribution, rather than fitting the cluster samples directly. It approximates the probability that vector x belongs to a particular class k_i as a sum of weighted Gaussian distributions centred at each cluster training sample. The output of the model is an estimate of the cluster class membership probabilities.

The architecture of the hybrid probabilistic neural networks in the proposed system is shown in Fig. 2. The network is composed of many interconnected processing units or neurons organized in successive layers. The hybrid probabilistic neural network for recognition of clusters of letters or words consists of five layers: cluster processing, cluster input, cluster pattern, summation and output layers. The cluster processing layer performs input value normalization of each value in the input vector. The cluster input layer unit does not perform any computation and simply distributes the input to the neurons in the next layer. In the pattern layer, there is one pattern neuron for each cluster training sample. Each pattern neuron forms a product of the weight vector w_j^i and the given cluster sample, where the weights entering a neuron are from a particular cluster sample. This product is then passed through the exponential activation function (2):

$$\exp\left(-\frac{\left(w_j^i - x\right)^T \left(w_j^i - x\right)}{2\sigma^2}\right) \tag{2}$$

It is necessary that both vectors x and w are normalized to unity. On receiving a pattern x from the cluster input layer, the neuron x_j^i of the cluster pattern layer computes its output (3):

$$\phi_j^i\left(x\right) = \frac{1}{\left(2\pi\right)^{s/2}\sigma^s}\exp\left(-\frac{1}{2\sigma^2}\left(x - x_j^i\right)^T\left(x - x_j^i\right)\right) \tag{3}$$

where s is the dimension of the cluster input pattern x, σ is a smoothing factor and x_j^i is the j-th training vector for the cluster patterns in class k_i. The superscript T denotes the transpose of the vector, and *exp* stands for the exponential function. The total number of the cluster pattern layer nodes is given as a sum of the cluster pattern units for all classes. The summation layer neurons compute the maximum likelihood of cluster pattern x being classified into k_i by summarizing and averaging the output of all neurons that belong to the same cluster class (4):

$$P_i\left(k_i\left|x\right.\right) = \frac{1}{\left(2\pi\right)^{s/2}\sigma^s}\frac{1}{N_i}\sum_{j=1}^{N_i}\exp\left(\frac{-\left(x - x_j^i\right)^T\left(x - x_j^i\right)}{2\sigma^2}\right) \tag{4}$$

where N_i is the number of cluster training patterns in class k_i. Eq. (4) is a sum of small multivariate Gaussian probability distributions that are centred at each cluster training sample. This function is used to generalize the classification

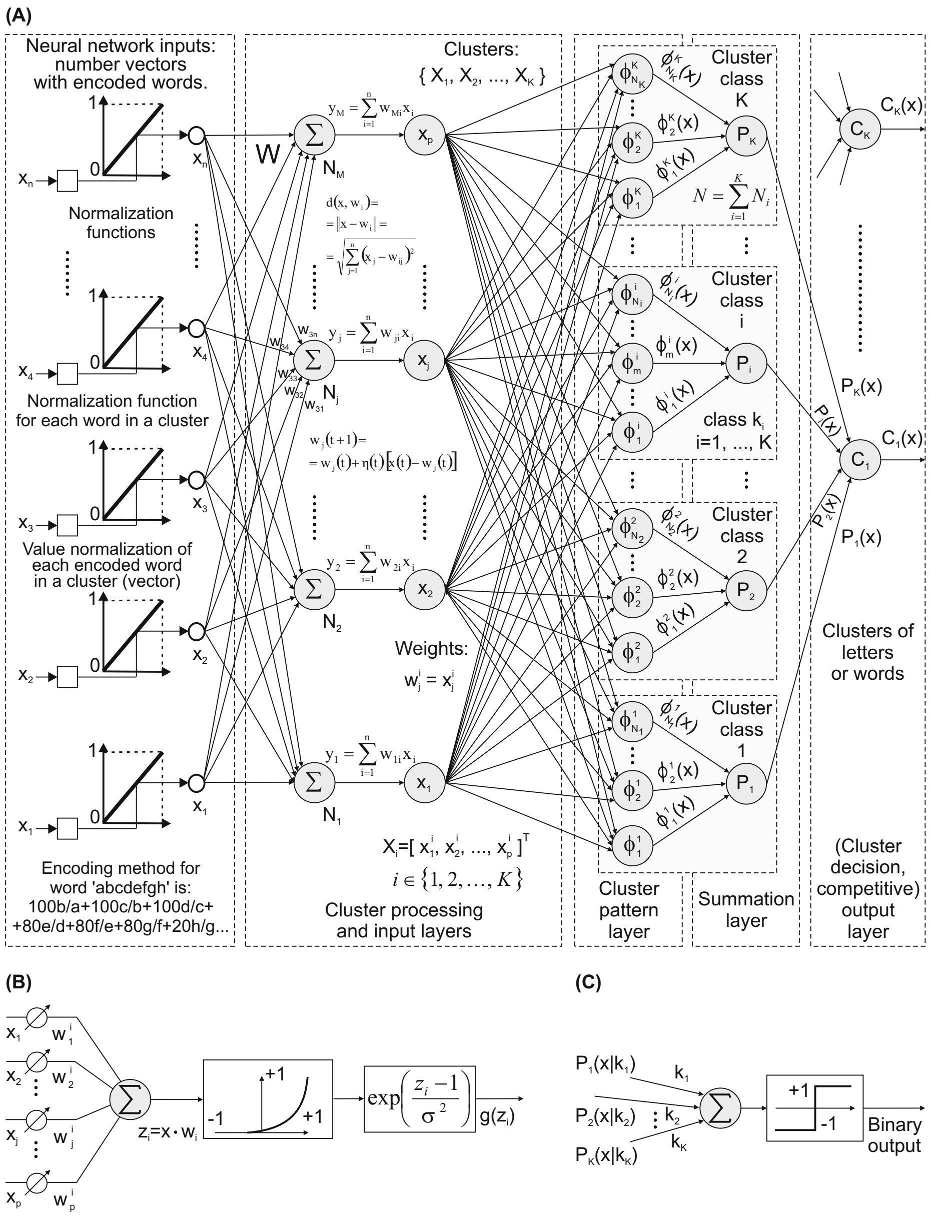

(A)
Neural network inputs: number vectors with encoded words.
Normalization functions
Normalization function for each word in a cluster
Value normalization of each encoded word in a cluster (vector)
Encoding method for word 'abcdefgh' is: 100b/a+100c/b+100d/c+ +80e/d+80f/e+80g/f+20h/g...
Clusters: { X1, X2, ..., XK }
W
d(x, wi) = ||x − wi|| = sqrt(Σ (xj − wij)^2)
wj(t+1) = wj(t) + η(t)[x(t) − wj(t)]
Weights: wij = xij
Xi=[xi1, xi2, ..., xip]T
i ∈ {1, 2, ..., K}
Cluster processing and input layers
Cluster class K
N = Σ Ni
Cluster class i
class ki i=1, ..., K
Cluster class 2
Cluster class 1
Cluster pattern layer
Summation layer
Clusters of letters or words
(Cluster decision, competitive) output layer
(B)
zi=x·wi
exp((zi − 1)/σ^2)
g(zi)
(C)
P1(x|k1)
P2(x|k2)
PK(x|kK)
Binary output

Fig. 2. (A) The hybrid probabilistic neural networks for recognition of clusters of letters or words, (B) Neuron of the pattern layer, (C) Neuron of the output layer to beyond the given cluster training samples. As the number of cluster training samples and their Gaussians increases the estimated probability density function approaches the true function of the cluster training set.

The classification decision for a cluster of letters or words is taken according to the inequality (5):

$$\begin{aligned}&\sum_{j=1}^{N_i} \exp\left(-\tfrac{1}{2\sigma^2}\left(x-x_j^i\right)^T\left(x-x_j^i\right)\right)\\ &\quad > \sum_{j=1}^{N_k} \exp\left(-\tfrac{1}{2\sigma^2}\left(x-x_j^k\right)^T\left(x-x_j^k\right)\right) \quad for\ \ all\ \ i\ \ and\ \ k\end{aligned} \tag{5}$$

Before classification, the sums in Eq. (5) are multiplied by their respective prior probabilities (P_i and P_k) calculated as the relative frequency of the cluster samples in each cluster class [18]. The decision layer classifies the cluster pattern x in accordance with the Bayes's decision rule based on the output of all the summation layer neurons using (6):

$$\begin{aligned}&\hat{C}(x) = \arg\ \max\left\{\tfrac{1}{(2\pi)^{s/2}\sigma^s}\tfrac{1}{N_i}\sum_{j=1}^{N_i}\exp\left(\tfrac{-\left(x-x_j^i\right)^T\left(x-x_j^i\right)}{2\sigma^2}\right)\right\}\\ &i = 1,\ 2,\ ...,\ K\end{aligned} \tag{6}$$

where $\hat{C}(x)$ denotes the estimated class of the cluster pattern x and K is the total number of classes in the cluster training samples [18].

The smoothing factor σ is the only factor that needs to be selected for training. A σ too small causes a very spiky approximation, which will not generalize clusters of letters or words well, whereas a σ too large will smooth out details of cluster structures. An appropriate σ is chosen empirically.

4 Experimental Results

Our corpus comprising 1,376 articles contains clusters of the types from 3 to 8 word clusters. The experimental results show the numbers of clusters in the corpus, which are presented in (Fig. 3A).

The proposed system allowed for recognition of any combination of meaningful word clusters with similar meanings but different lexico-grammatical patterns. The tests measured the performance of the cluster meaning recognition. The effectiveness of the system was achieved to a satisfactory level. As shown in Fig. 3B, the ability of the hybrid probabilistic neural network to recognize a cluster depends on the number of words in that cluster. For best performance, the neural network requires a minimum number of words of each cluster to be recognized as its input.

Important factors are both the neural network design (i.e., selection of the smoothing factor (σ)) and development of representative training patterns of word clusters by the proposed system.

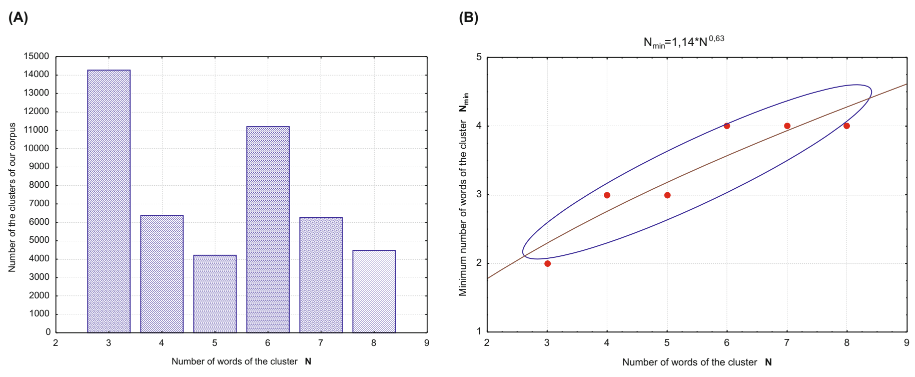


Fig. 3. (A) number of clusters of our corpus vs. number of words of the cluster, (B) sensitivity of word cluster meaning recognition: minimum number of words of the cluster being recognized vs. number of cluster component words

5 Conclusions and Perspectives

It is assumed that language processing is closely tied to a user's experience, and that distributional frequencies of words and structures play an important role in learning. Therefore the interest in the statistical profile of language usage plays an important role in research. This paper has developed a method which allows for extraction of possible word cluster components in a corpus for training hybrid probabilistic neural networks. The networks are capable of recognizing word clusters with similar meaning but different lexico-grammatical patterns. It has long been an ambition of corpus linguistics to investigate fully relationships between form and meaning, sense and syntax [17]. The patterns of language have been revealed by corpus linguistics through concordance lines, word clusters, collocation and colligation but there is no automated way of generating these word clusters. It might be useful for corpus linguistics to learn from neural networks how to generate word clusters automatically based on the training of the aforementioned networks with corpus examples and thereby bridge the gap between data-driven Hallidayan approaches to language and the more formalized Chomskyan predictive approach.

References

1. Biber, D.: Variation across speech and writing. Cambridge University Press, Cambridge (1988)
2. Biber, D.: Using register-diversified corpora for general language studies. Computational Linguistics 19(2), 219–241 (1993)
3. Biber, D., Conrad, S., Reppen, R.: Corpus linguistics: Investigating language structure and use. Cambridge University Press, Cambridge (1998)
4. Cacoullous, R.: Estimation of a probability density. Annals of the Institute of Statistical Mathematics (Tokyo) 18(2), 179–189 (1966)

5. Carter, R., Hughes, R., McCarthy, M.: Exploring grammar in context. Cambridge University Press, Cambridge (2000)
6. Jurafsky, D.: A probabilistic model of lexical and syntactic access and disambiguation. Cognitive Science 20(2), 137–194 (1996)
7. Jurafsky, D., Martin, J.H.: Speech and language processing: An introduction to natural language processing. In: Speech Recognition, and Computational Linguistics. Prentice-Hall, New Jersey (2000)
8. Kacalak, W., Stuart, K., Majewski, M.: Intelligent natural language processing. In: Jiao, L., Wang, L., Gao, X.-b., Liu, J., Wu, F. (eds.) ICNC 2006. LNCS, vol. 4221, pp. 584–587. Springer, Heidelberg (2006)
9. Kacalak, W., Stuart, K., Majewski, M.: Selected problems of intelligent handwriting recognition. Advances in Soft Computing 41, 298–305 (2007)
10. Kennedy, G.: An introduction to corpus linguistics. Longman, London (1998)
11. Lapata, M., Keller, F., Schulte, S.: Verb frame frequency as a predictor of verb bias. Journal of Psycholinguistic Research 30(4), 419–435 (2001)
12. Manning, C., Schtze, H.: Foundations of statistical natural language processing. MIT Press, Cambridge (1999)
13. Murthy, V.K.: Estimation of a probability density. The Annals of Mathematical Statistics 36(3), 1027–1031 (1965)
14. Parzen, E.: On estimation of a probability density function and mode. The Annals of Mathematical Statistics 33(3), 1065–1076 (1962)
15. Roland, D., Elman, J.L., Ferreira, V.S.: Why is that? Structural prediction and ambiguity resolution in a very large corpus of English sentences. Cognition 98(3), 245–272 (2006)
16. Sampson, G.: English for the computer. Oxford University Press, Oxford (1995)
17. Sinclair, J.: Corpus, Concordance, Collocation. Oxford University Press, Oxford (1991)
18. Specht, D.F.: Probabilistic neural networks. Neural Networks 3(1), 109–118 (1990)
19. Specht, D.F.: A general regression neural network. IEEE Transactions on Neural Networks 2(6), 568–576 (1991)
20. Specht, D.F.: Enhancements to probabilistic neural networks. In: Proceedings of the IEEE International Joint Conference on Neural Networks, Baltimore Maryland USA, vol. 1, pp. 761–768 (1992)
21. Specht, D.F., Romsdahl, H.: Experience with adaptive probabilistic neural networks and adaptivegeneral regression neural networks. In: IEEE World Congress on Computational Intelligence, IEEE International Conference on Neural Networks, Orlando Florida USA, vol. 2, pp. 1203–1208 (1994)
22. Stuart, K., Majewski, M.: Selected problems of knowledge discovery using artificial neural networks. In: Liu, D., Fei, S., Hou, Z., Zhang, H., Sun, C. (eds.) ISNN 2007. LNCS, vol. 4493, pp. 1049–1057. Springer, Heidelberg (2007)
23. Stuart, K., Majewski, M.: A new method for intelligent knowledge discovery. Advances in Soft Computing 42, 721–729 (2007)
24. Stuart, K., Majewski, M.: Artificial creativity in linguistics using evolvable fuzzy neural networks. In: Hornby, G.S., Sekanina, L., Haddow, P.C. (eds.) ICES 2008. LNCS, vol. 5216, pp. 437–442. Springer, Heidelberg (2008)
25. Stuart, K., Majewski, M.: Evolvable neuro-fuzzy system for artificial creativity in linguistics. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 46–53. Springer, Heidelberg (2008)
26. Stuart, K.D., Majewski, M., Trelis, A.B.: Selected problems of intelligent corpus analysis through probabilistic neural networks. In: Zhang, L., Lu, B.-L., Kwok, J. (eds.) ISNN 2010. LNCS, vol. 6064, pp. 268–275. Springer, Heidelberg (2010)

Decision-Making in Drosophila with Two Conflicting Cues

Kuijie Cai[1,2], Jihong Shen[2], and Si Wu[1]

[1] Institute of Neuroscience, Shanghai Institutes for Biological Sciences, Chinese Academy of Sciences, Shanghai 200031, China
[2] College of Science, Harbin Engineering University, Harbin 150001, China
{kjcai,siwu}@ion.ac.cn, shenjihong@hrbeu.edu.cn

Abstract. Experimental data has revealed that the decision behavior of Drosophila is almost linear when facing a single cue. However, when two conflicting cues are presented, the decision behavior of Drosophila becomes winner-takes-all. We propose a connectionist model to elucidate the underlying computational mechanism. We consider two neural states, representing different action choices, compete with each other through mutual inhibition. They receive inputs from Mushroom Body when conflicting information arise. The role of Mushroom Body is to average out temporal noises in external inputs, so that subtle differences between two conflicting cues can be identified, leading to higher discrimination accuracies. Our model successfully describes the experimental findings.

Keywords: decision-making, connectionist model, conflict, feedback, gating mechanism.

1 Introduction

Decision-making refers to the behavior that a subject makes a choice when facing multiple alternatives. Decision-making is a fundamental cognitive function of the brain, which guides our daily activities, for instances, we make choices on food in a restaurant, on clothes when going to a party, and on routes in traveling. Understanding the mechanism of decision-making is critical for us to understand high-level brain functions.

Over the past decades, a large volume of research has been devoted to study the behavior natures and the neural correlates of decision-making in both human and animal models, but little progress has been made on the circuit-level computational mechanism underlying decision-making. A promising model was proposed by Wang et al., who suggested that two neural groups biased for different choices compete with each other through mutual inhibition to make the final decision [1]. This model has successfully predicted several aspects of the decision-making behaviors of rhesus monkeys in a motion perception task.

In a recent experiment study, Zhang et al. explored the decision-making behavior of Drosophila and found very interesting phenomena [2]. They trained flies to make a decision of either following an upper visual bar or a lower one

© Springer-Verlag Berlin Heidelberg 2011

depending on the cues. Two visual cues are used, one is the elevation of bars and the other the color of bars (blue or green). They measured the choice behavior of a fly by a preference index (PI), which is defined by the difference between the fractions of time the fly following the upper or the lower bar. They found that: 1) when a single cue (the elevation of bars) is presented, the choice curve of the fly is linear, that is, PI increases almost linearly with the cue strength (the latter is measured by the elevation difference between two bars); and 2) when two conflicting cues are presented (i.e., the elevation cue favors the upper bar, whereas, the color cue the lower one), the choice curve of the fly displays a sigmoid shape, indicating that the fly's decision is winner-takes-all.

In this work, we propose a network model to elucidate the computational mechanism underlying the above observed interesting behaviors. In particular, we aim to answer two questions: 1) why the choice curve of a fly is linear when facing a single cue; and 2) why the choice behavior becomes winner-takes-all when two conflicting cues are presented. We hope this study will give us insight into understanding some general principles of decision-making in the brain.

2 The Model

2.1 The Network Architecture

Based on the known experimental data, we propose the following neural architecture for implementing decision-making in Drosophila (see Fig. 1). The stimulus information encoding the elevation and/or the color of bars is first extracted in the visual system of Drosophila. It was found that the short-term memory for elevation of visual inputs is localized in the fan-shaped body of the central complex [3]. This visual information is propagated to the decision-making layer for generating motor commands. Two possible motor commands may be generated, one is to follow the upper bar and the other the lower one. The two choices are competing with each other depending on the training protocol and the input cues, and finally a biased decision is reached. Experimental studies have revealed that Mushroom Body (MB) in Drosophila is actively involved in the decision-making process [2]. Blocking MB would made the choice curve of Drosophila in the conflicting-cue task lose its winner-takes-all nature. We therefore assume a connection between the decision-making layer and MB exists. Furthermore, experimental data showed that the impact of MB on decision-making is modulated by dopaminergic neurons. It has been suggested that the dopamine-MB circuit mediates the winner-takes-all decision behavior in the presence of conflicting cues [2]. A related evidence has been found recently, that is, the relative salience of learned odor cues is modulated by gating MB through the MB-MP (dopaminergic) neurons [4]. Therefore, we suggest that dopaminergic neurons may send tonic inhibitory inputs to MB, suppressing its connection to the decision-making layer in normal situations, and this suppression is released when conflicting information is observed. Similar gating mechanisms implemented by dopamine have also been widely reported in rat, monkey and human research [5][6][7]. In the studies with human and monkeys, it has been observed that anterior cingulate cortex (ACC) serves to detect conflicting information of external inputs [8][9]. We propose that a similar neural substrate exists in Drosophila, hereafter we call it conflict monitor. The conflict monitor may achieve its function by inhibiting the activity of dopaminergic neurons when conflicting cues arise, consequently the gate for the MB output to the decision-making layer is opened.

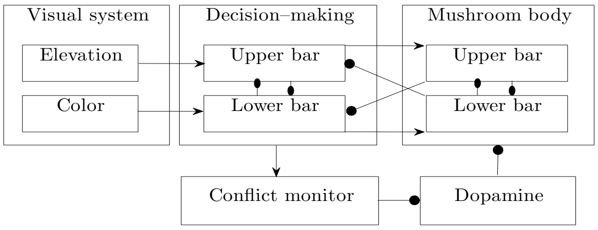


Fig. 1. The possible neural architecture for implementing decision-making in Drosophila. The elevation and/or the color information is extracted in the visual system. This information is subsequently propagated to the decision-making layer for generating the associated motor commands. The decision-making layer is connected with MB. However, the opening of this connection is modulated by dopaminergic neurons, and the latter is further controlled by a conflict monitor.

2.2 The Network Dynamics

Since the fine structure of neural circuits in Drosophila involved in decision-making is largely unknown, we employ a simple connectionist model to describe the behaviors of the above network dynamics. A large volume of theoretical study has demonstrated that connectionist models can successfully elucidate the key mechanisms underlying many high-level cognitive functions including decision-making (see, e.g., [10][11][12]).

In particular, we use two variables, z_1 and z_2, to denote the neural states in the decision-making layer. They compete with each other through mutual inhibition. The consequence $z_1 > z_2$ means that the network chooses to follow the upper bar, and $z_1 < z_2$ the opposite choice.

We consider no conflicting cues first. No input from MB to the decision-making layer is available. The dynamics of z_i, for $i = 1, 2$, is governed by a leakage term, a mutual inhibition term, and a biased input. The dynamics of z_i may be written as

$$\tau_z \frac{dz_1}{dt} = -z_1 - w_z z_2 + I_0 + I_b + \eta_1, \quad (1)$$

$$\tau_z \frac{dz_2}{dt} = -z_2 - w_z z_1 + I_0 + \eta_2, \quad (2)$$

where τ_z is the time constant of the decision-making layer. I_0 is the non-biased input to the two neural groups, $I_0 > 0$ implying that a visual bar is a natural attractive stimulus to Drosophila. Without loss of generality, we consider z_1

receives the biased input I_b, $I_b > 0$ implies that a fly tends to follow the upper bar. The variables, η_i, for $i = 1, 2$, denote gaussian white noise of zero mean and with variances $D(\eta_1) = I_0 + I_b$ and $D(\eta_2) = I_0$ (according to the Poisson statistics of neural firing). The input noise is the resource for a fly generating randomness in decision-making. The mutual inhibition term, $-w_z z_i$, for $i = 1, 2$, represents the competition between two neural states (we may alternatively use a nonlinear interaction term, $-w_z f(z)$ with $f(z)$ a sigmoid function, to implement the mutual inhibition [11]. However, we find that in our model a linear inhibition mechanism is already sufficient to justify the experimental behaviors).

In the presence of conflicting cues, the feedback input from MB to the decision-making layer is activated, and the network dynamics is written as

$$\tau_z \frac{dz_1}{dt} = -z_1 - w_z z_2 + I_0 + I_b^1 - k_z m_2 + \eta_1, \tag{3}$$

$$\tau_z \frac{dz_2}{dt} = -z_2 - w_z z_1 + I_0 + I_b^2 - k_z m_1 + \eta_2, \tag{4}$$

$$\tau_m \frac{dm_1}{dt} = -m_1 - w_m m_2 + k_m z_1, \tag{5}$$

$$\tau_m \frac{dm_2}{dt} = -m_2 - w_m m_1 + k_m z_2, \tag{6}$$

where τ_m is the time constant of MB. The terms, $k_z m_i$, for $i = 1, 2$, are the feedback inputs from MB to the decision-making layer, and $k_m z_i$ are similarly defined. Now, the decision-making layer receives two biased inputs, I_b^1 and I_b^2, which represent the two competitive choices induced by two conflicting cues. The competition between z_1 and z_2 mediated by MB causes the network to make the winner-takes-all decision.

We consider the condition $\tau_m \gg \tau_z$. To see the contribution of MB clearly, let us consider first $w_m = 0$ and $k_m = 1$. Thus, m_i is determined by the average value of z_i over a period τ_d. This largely averages out the fluctuations in z_i. Subsequently, the value of m_i feedback to the dynamics of z_i, which helps the network to diminish the disturbance of noises on decision-making. We will confirm this point by simulation.

3 Results

3.1 The Case of a Single Cue

We first solve the network dynamics in the presence of a single cue. Let us define $z = z_1 - z_2$. Subtracting Eq.(1) by (2), we obtain

$$\tau_m \frac{dz}{dt} = -(1 - w_z)z + I_b + \eta, \tag{7}$$

where η is gaussian white noise of zero mean and variance $D(\eta) = 2I_0 + I_b$ (note η_1 and η_2 are independent to each other). This is the standard Ornstein-Uhlenbeck process, whose solution is calculated to be

$$z(t) = z(0)e^{-(1-w_z)t/\tau_z} + \frac{I_b}{1-w_z}\left[1 - e^{-(1-w_z)t/\tau_z}\right] + \sqrt{(2I_0 + I_b)/\tau_z}e^{-(1-w_z)t/\tau_z}\int_0^t e^{-(1-w_z)t'/\tau_z}dB_{t'}, \qquad (8)$$

where $z(0)$ is the initial value of $z(t)$ and B_t denotes the standard Brownian process.

The mean and the variance of $z(t)$ are given by

$$E[z(t)] = z(0)e^{-(1-w_z)t/\tau_z} + \frac{I_b}{1-w_z}\left[1 - e^{-(1-w_z)t/\tau_z}\right], \qquad (9)$$

$$D[z(t)] = \frac{2I_0 + I_b}{2(1-w_z)}\left[1 - e^{-2(1-w_z)t/\tau_z}\right]. \qquad (10)$$

We consider a fly reaches a decision when t is sufficiently large. Since τ_z is very small in practice (in the order of mini-second according to the neural dynamics), we can effectively take the limit of $t \to \infty$ to compute the distribution of $z(t)$. In the limit $t \to \infty$, z satisfies a gaussian distribution given by

$$p(z) = \sqrt{\frac{1-w_z}{\pi(2I_0 + I_b)}}\exp\left[-\frac{1-w_z}{2I_0 + I_b}(z - \frac{I_b}{1-w_z})^2\right]. \qquad (11)$$

The probability $P(z > 0)$ is the fraction of time for a fly choosing to follow the upper bar, and $P(z < 0)$ the lower bar. Thus, the performance index PI of the fly is calculated to be

$$\begin{aligned} PI &= P(z > 0) - P(z < 0), \\ &= 2\int_0^\infty p(z)dz - 1. \end{aligned} \qquad (12)$$

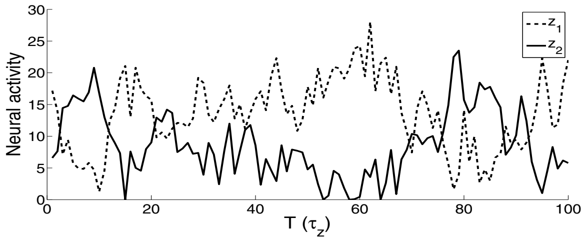
From eqs.(11), we see that both the mean and variance of $p(z)$ increases with I_b. Their joint effects cause PI increases almost linearly with I_b (see Fig. 2b).

We carry out simulation to further confirm the above analysis. Fig. 2(a) shows the typical performance of the network in a single trial. We see that the network state may randomly switch between two choices, leading to the stochastic behavior in decision-making. The network choice, quantified by PI counted over a sufficiently long time, increases linearly with the biased input (see Fig. 2(b)). This agrees well with the experimental finding.

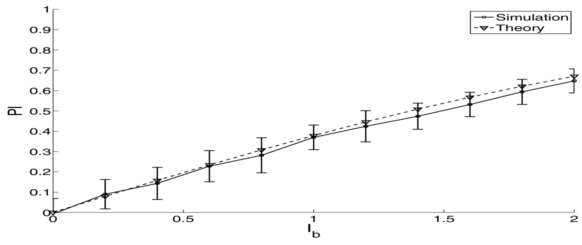
3.2 The Case of Conflicting Cues

We now consider the case of two conflicting cues. Define $z = z_1 - z_2$, and $m = m_1 - m_2$. Subtracting Eq.(3) by (4), Eq.(5) by (6), we obtain

$$\tau_z\frac{dz}{dt} = -(1-w_z)z + I_b^1 - I_b^2 + k_z m + \eta, \qquad (13)$$



(a)



(b)

Fig. 2. (a) The typical behavior of the network dynamics in a single trial when a single cue is presented. The parameters are $\tau_z = 1, w_z = 0.8, I_0 = 20, I_b = 0.8$. (b) The choice curve: PI vs. I_b. Each PI value is measured in a time window $1000\tau_z$ after the network reaching a stationary state. The final result is obtained by averaging over 500 trials (solid line). The theoretical analysis is shown by dashed line. The same parameters are used as in (a).

$$\tau_m \frac{dm}{dt} = -(1 - w_m)m + k_m z, \tag{14}$$

where η is gaussian white noise of zero mean and variance $D(\eta) = 2I_0 + I_b^1 + I_b^2$. Rewrite the two above equations in the two dimensional Itô form:

$$d\mathbf{X} = \begin{pmatrix} dm \\ dz \end{pmatrix} = \mathbf{A} \cdot \mathbf{X}dt + \mathbf{B}dt + \mathbf{C}dB_t, \tag{15}$$

where $\mathbf{A} = \begin{pmatrix} -(1 - w_m)/\tau_m & k_m/\tau_m \\ k_z/\tau_z & -(1 - w_z)/\tau_z \end{pmatrix}, \mathbf{X} = \begin{pmatrix} m \\ z \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 0 \\ (I_b^1 - I_b^2)/\tau_z \end{pmatrix},$
$\mathbf{C} = \begin{pmatrix} 0 \\ \sqrt{(2I_0 + I_b^1 + I_b^2)/\tau_z} \end{pmatrix}.$

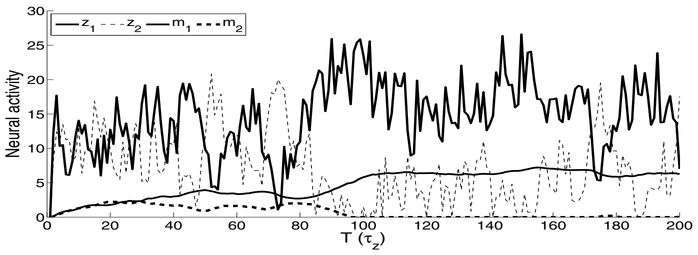
$\mathbf{X}(t)$ is calculated to be:

$$\mathbf{X}(t) = exp(\mathbf{A}t) \left[\mathbf{X}(0) + \int_0^t exp(-\mathbf{A}t') \cdot \mathbf{B}dt' + \int_0^t exp(-\mathbf{A}t') \cdot \mathbf{C}dB_t' \right]. \tag{16}$$

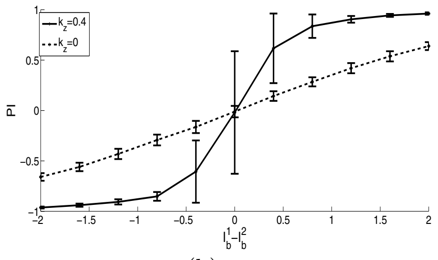
In the below we carry out simulation to describe the network behavior.

Fig. 3(a) shows the typical performance of the network in a single trial when two conflicting cues are presented. We see that in the initial period, z_1 and z_2

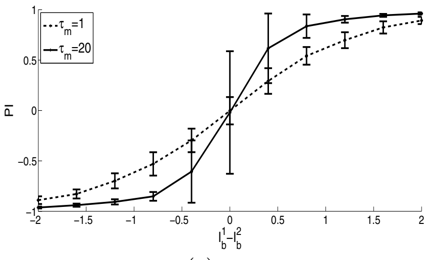
are only vaguely separated. The discrepancy between m_1 and m_2 is gradually built up in the time order of τ_m. Since m_i is determined by the temporal average of z_i, it has smaller fluctuations than z_i. After a sufficiently long time, z_1 and z_2 become clearly separated due to the feedback contribution of m_i. Fig. 3(b) shows that the choice curve: PI vs. $(I_b^1 - I_b^2)$ with I_b^2 fixed. As expected, PI increases with $(I_b^1 - I_b^2)$. We further observe that the slope of the choice curve increases with the feedback strength k_z. This is understandable, since the contribution of MB is to diminish the disturbance of noises. When k_z is sufficiently large, PI increases sharply with $(I_b^1 - I_b^2)$ but saturates to the maximum value of 1 (or decreases to the minimum value -1). This makes the choice curve exhibit a sigmoid shape. In behavior, the Drosophila's decision behaves as if winner-takes-all. In Fig. 3(c), we further study the impact of τ_m on decision-making. We see that the slope of the choice curve increases with τ_m. This is understandable. Since m_i is to average out fluctuations in z_i over a period τ_m. The larger the value of τ_m, the more efficient noise are diminished. Therefore, larger τ_m leads to higher accuracy in distinguishing I_b^1 from I_b^2.



(a)



(b)



(c)

Fig. 3. (a) The typical performance of the network in a single trial when two conflicting cues are presented. The parameters are $\tau_z = 1, \tau_m = 20, w_z = w_m = 0.8, k_z = k_m = 0.4$. (b) The choice curves with different feedback strength k_z. (c) The choice curves with different τ_m.

4 Conclusions

In the present study, we have built up a network model to elucidate the computational mechanism underlying the decision-making performance of Drosophila when facing two conflicting cues. We consider there exists a neural substrate monitoring the conflicting information, which activates the Dopamine-MB circuit in the presence of conflicting choices. The output of MB is then feedback to the decision-making layer. In computation, the role of MB is to average out the temporal fluctuations in external inputs, so that the subtle difference between the two cues is distinguished. In behavior, the contribution of MB causes Drosophila to make winner-takes-all decision. Our model successfully describes the experimental findings.

References

1. Wang, X.: Probabilistic decision making by slow reverberation in cortical circuits. Neuron 36(5), 955–968 (2002)
2. Zhang, K., Guo, J., Peng, Y., Xi, W., Guo, A.: Dopamine-mushroom body circuit regulates saliency-based decision-making in Drosophila. Science's STKE 316(5833), 1901 (2007)
3. Liu, G., Seiler, H., Wen, A., Zars, T., Ito, K., Wolf, R., Heisenberg, M., Liu, L.: Distinct memory traces for two visual features in the Drosophila brain. Nature 439(7076), 551–556 (2006)
4. Krashes, M., DasGupta, S., Vreede, A., White, B., Armstrong, J., Waddell, S.: A neural circuit mechanism integrating motivational state with memory expression in Drosophila. Cell 139(2), 416–427 (2009)
5. O'Reilly, R., Frank, M.: Making working memory work: a computational model of learning in the prefrontal cortex and basal ganglia. Neural Computation 18(2), 283–328 (2006)
6. Baier, B., Karnath, H., Dieterich, M., Birklein, F., Heinze, C., Muller, N.: Keeping Memory Clear and Stable–The Contribution of Human Basal Ganglia and Prefrontal Cortex to Working Memory. Journal of Neuroscience 30(29), 9788 (2010)
7. Romanides, A., Duffy, P., Kalivas, P.: Glutamatergic and dopaminergic afferents to the prefrontal cortex regulate spatial working memory in rats. Neuroscience 92(1), 97–106 (1999)
8. Botvinick, M., Cohen, J., Carter, C.: Conflict monitoring and anterior cingulate cortex: an update. Trends in Cognitive Sciences 8(12), 539–546 (2004)
9. Van Veen, V., Carter, C.: The anterior cingulate as a conflict monitor: fMRI and ERP studies. Physiology & Behavior 77(4-5), 477–482 (2002)
10. Usher, M., McClelland, J.: The time course of perceptual choice: The leaky, competing accumulator model. Psychological Review-New York 108(3), 550–592 (2001)
11. Liu, Y., Holmes, P., Cohen, J.: A neural network model of the Eriksen task: Reduction, analysis, and data fitting. Neural Computation 20(2), 345–373 (2008)
12. Gao, J., Wong-Lin, K., Holmes, P., Simen, P., Cohen, J.: Sequential effects in two-choice reaction time tasks: Decomposition and synthesis of mechanisms. Neural Computation 21(9), 2407–2436 (2009)

Time-Varying Quadratic Programming by Zhang Neural Network Equipped with a Time-Varying Design Parameter $\gamma(t)$

Zhan Li and Yunong Zhang*

School of Information Science and Technology
Sun Yat-sen University, Guangzhou 510006, China
zhynong@mail.sysu.edu.cn

Abstract. In this paper, a recurrent neural network termed Zhang neural network (ZNN) with a time-varying design parameter $\gamma(t)$ is developed and presented to solve time-varying quadratic programs subject to time-varying linear equalities. The updated design formula for the ZNN model possesses more generality because the design parameter considered is actually (e.g., in hardware implementation) time-varying, i.e., $\gamma(t)$. The state vector of such a ZNN model with time-varying design parameter $\gamma(t)$, can also globally exponentially converge to the theoretical optimal solution pair of the time-varying linear-equality-constrained quadratic program. To achieve superior convergence of the ZNN model, nonlinear activation functions are adopted as well, as compared with the linear-activation-function case. Simulation results substantiate the efficiency of such a ZNN model with a time-varying design parameter $\gamma(t)$ aforementioned.

Keywords: Time-varying, Quadratic program, Neural network, Global convergence.

1 Introduction

The problem of linear-equality-constrained quadratic program is widely encountered in many important applications [1, 2]. It can be considered as an approximate avenue for solving the nonlinear optimization problem. Generally speaking, there are two main types of solution to the problem of the linear-equality-constrained quadratic program problem. The traditional one can be generalized as the serial-processing numerical algorithms with the minimal arithmetic operations proportional to the cube of the coefficient matrix dimension. However, such serial-processing numerical algorithms may not be efficient enough for large-scale online (or real-time) applications due to higher computational complexity [3].

Being the other important type of solution to linear-equality-constrained quadratic programs and relevant optimization problems solving, many parallel processing computational methods, based on artificial analog models, have

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

been proposed, analyzed, and implemented on specific architectures, e.g., the analog and neural-dynamic solvers [4–9]. Such a neural-dynamic approach is now regarded as a powerful alternative to online computation and optimization, owing to its potential parallel-processing distributed nature and convenience of hardware implementation [1, 4, 11].

However, in the most past literature, the linear-equality-constrained quadratic optimization problem [3] is considered and investigated with static/constant coefficients. The conventional computational schemes (e.g., the serial-processing method and gradient-based neural-dynamic methods) may be efficient for solving such linear-equality-constrained quadratic optimization problems, but may be less favorable for the time-varying linear-equality-constrained quadratic program problem (where the coefficients of the objective function and linear equality constraints are time-varying). Recently, a new recurrent neural network termed Zhang neural network (ZNN) has been proposed and exploited as a powerful alternative for solving such a time-varying linear-equality-constrained quadratic program problem [8, 9]. Differing from the conventional gradient-based neural network (GNN) model, such a ZNN model makes full use of the first-order time-derivative information of the time-varying coefficients, and forces the unbounded/indefinite time-varying matrix-valued error function to decrease to zero rapidly during its solving process.

The proposed ZNN model in [9] for solving the time-varying linear-equality-constrained quadratic program problem is with a constant design parameter $\gamma > 0$, which scales the convergence rate during the solution process. Such a ZNN model can achieve the global exponential convergence to the theoretical solutions of the time-varying linear-equality-constrained quadratic programs. The design parameter γ, being a set of reciprocals of capacitance-parameters, should possibly be time-varying [i.e., $\gamma(t)$] in practice for potential analog/digital circuits implementation. Motivated by this point, in the paper, we show that the global exponential convergence property of the ZNN model equipped with a time-varying design parameter $\gamma(t)$ can still be guaranteed for solving online the time-varying linear-equality-constrained quadratic programs.

2 Problem Formulation and Neural-Network Solvers

In this paper, let us consider the following time-varying strictly-convex quadratic program subject to time-varying linear-equality constraints:

$$\begin{aligned} \text{minimize} \quad & x^T(t)P(t)x(t)/2 + q^T(t)x(t), \\ \text{subject to} \quad & A(t)x(t) = b(t), \end{aligned} \tag{1}$$

where matrices $P(t) \in R^{n\times n}$ and $A(t) \in R^{m\times n}$ are positive definite and of full rank respectively at any time instant $t \in [0, +\infty)$, $q(t) \in R^n$ and $b(t) \in R^m$ are time-varying vector coefficients, $x(t) \in R^n$ is the unknown vector to be optimized. In addition, the coefficient matrices and vectors, together with their first-order time-derivatives, are assumed to be known analytically or estimated accurately. It is worth noting that the optimal solution $x^*(t) \in R^n$ in problem (1) to be solved is always time-varying, rather than a time-invariant (or to say, static, constant) optimal solution $x^* \in R^n$ usually considered in static quadratic optimization problems [3].

2.1 ZNN Model

To monitor the solving process of time-varying quadratic program (1), the following Zhang function is defined (which is a vector-valued indefinite and lower-unbounded error function, rather than the scalar-valued norm-based nonnegative energy functions usually used in gradient-based neural network approaches):

$$e(t) = W(t)y(t) + u(t) \in R^{n+m},$$

with

$$W(t) = \begin{bmatrix} P(t) & A^T(t) \\ A(t) & \mathbf{0} \end{bmatrix}, y(t) = \begin{bmatrix} x(t) \\ \lambda(t) \end{bmatrix}, u(t) = \begin{bmatrix} q(t) \\ -b(t) \end{bmatrix} \in R^{n+m},$$

where $\lambda(t) \in R^m$ denotes the Lagrange-multiplier vector. To make each entry of $e(t) \in R^{n+m}$ converge to zero, the following ZNN design formula has been exploited [7–9]:

$$\dot{e}(t) = -\gamma \mathcal{F}(e(t)), \tag{2}$$

where design parameter $\gamma > 0$, being a set of reciprocals of capacitance parameters, should be set as large as the hardware would permit, or set appropriately for simulative purposes [11]. $\mathcal{F}(\cdot) : R^{n+m} \to R^{n+m}$ denotes an activation-function processing-array. In addition, each scalar-valued processing-unit $f(\cdot)$ of $\mathcal{F}(\cdot)$ should be a monotonically-increasing odd activation function. Since March 2001 [6], we have introduced and investigated five types of activation functions (i.e., linear activation function, power activation function, power-sum activation function, sigmoid activation function and power-sigmoid activation function) [6–8, 10]. Other types of activation functions can then be generalized by understanding the above five basic types of activation functions.

In this paper, for more practical purposes, we modify the ZNN design formula (2) into the general form as follows:

$$\dot{e}(t) = -\gamma(t)\mathcal{F}(e(t)). \tag{3}$$

Differing from design formula (2), new design formula (3) possesses more generality, since design parameter $\gamma(t)$ is time-varying, which is more suitable for real physical implementation environment for the ZNN model (i.e., capacitance parameters may actually change with time in most analog circuits [11]).

Expanding equation (3), we obtain the following ZNN model depicted in a nonlinear implicit dynamic equation:

$$W(t)\dot{y}(t) = -\dot{W}(t)y(t) - \gamma(t)\mathcal{F}(W(t)y(t) + u(t)) - \dot{u}(t). \tag{4}$$

If the design parameter $\gamma(t) := \gamma > 0$ is constant and independent of time $t \in [0, +\infty)$, the state vector $x(t) \in R^n$ of ZNN model (4) can globally exponentially converge to the unique time-varying optimal solution $x^*(t) \in R^n$ of time-varying quadratic program (1) [9]. However, if the design parameter $\gamma(t)$ is a time-varying term, such convergence property may not be sure for ZNN (4). In the ensuing propositions, we show that, as long as the design parameter $\gamma(t)$ is time-varying under certain conditions, global exponential convergence can be achieved for ZNN model (4) as well.

Proposition 1. *Consider time-varying strictly-convex quadratic program (1). If there exists a positive scalar $\epsilon > 0$ such that $\gamma(t) \geq \epsilon$, state vector $x(t) \in R^n$ of ZNN model (4), starting from any initial state $x(0) \in R^n$, globally converges to the unique theoretical time-varying optimal solution $x^*(t) \in R^n$ of (1).*

According to Proposition 1, to make global convergence of ZNN model (4) to optimal solution of (1), we can choose the following categories of $\gamma(t)$:

1) positive constant scalar, $\gamma(t) = c > 0$;
2) power term, $\gamma(t) = t^\rho + c$ where $\rho > 0$ and $c > 0$;
3) polynomial term, $\gamma(t) = \sum_{i=0}^N a_i t^i$ where $a_i > 0$;
4) exponential term, $\gamma(t) = \exp(\zeta t) + c$ where $\zeta > 0$ and $c > 0$; and,
5) other forms of $\gamma(t)$ which is larger than or equal to c where $c > 0$.

Proposition 2. *If there exists a positive scalar $\epsilon > 0$ such that $\gamma(t) \geq \epsilon$, starting from any initial condition $x(0) \in R^n$, state vector $x(t) \in R^n$ of ZNN model (4) activated by linear functions globally exponentially converges to the theoretical time-varying optimal solution $x^*(t) \in R^n$ of (1).*

Proposition 3. *If there exists a positive scalar $\epsilon > 0$ such that $\gamma(t) \geq \epsilon$, starting from any initial state $x(0) \in R^n$, the state vector $x(t) \in R^n$ of ZNN model (4) activated by power-sigmoid functions*

$$f(v) = \begin{cases} v^\rho, & |v| \geqslant 1, \\ \frac{1+\exp(-\xi)}{1-\exp(-\xi)} \cdot \frac{1-\exp(-\xi v)}{1+\exp(-\xi v)}, & |v| \leqslant 1, \end{cases}$$

with suitable design parameters (e.g., odd integer $\rho \geqslant 3$ and $\xi \geqslant 2$), superiorly converges to the theoretical time-varying optimal solution $x^*(t) \in R^n$ of (1), as compared with the situation of Proposition 2.

Proposition 4. *If there exists a positive scalar $\epsilon > 0$ such that $\gamma(t) \geq \epsilon$, starting from any initial state $x(0) \in R^n$, $x(t) \in R^n$ of ZNN model (4) activated by power-sum functions $f(v) = \sum_{k=1}^N v^{2k-1}$ superiorly converges to the theoretical optimal solution $x^*(t) \in R^n$ of (1), as compared with the situation of Proposition 2.*

Proposition 5. *If there exists a positive scalar $\epsilon > 0$ such that $\gamma(t) \geq \epsilon$, starting from any initial state $x(0) \in R^n$, the state vector $x(t) \in R^n$ of ZNN model(4) activated by hyperbolic sine functions $f(v) = exp(\xi v)/2 - exp(-\xi v)/2$ with parameter $\xi \geqslant 1$ superiorly converges to the theoretical time-varying optimal solution $x^*(t) \in R^n$ of (1), as compared with the situation of Proposition 2.*

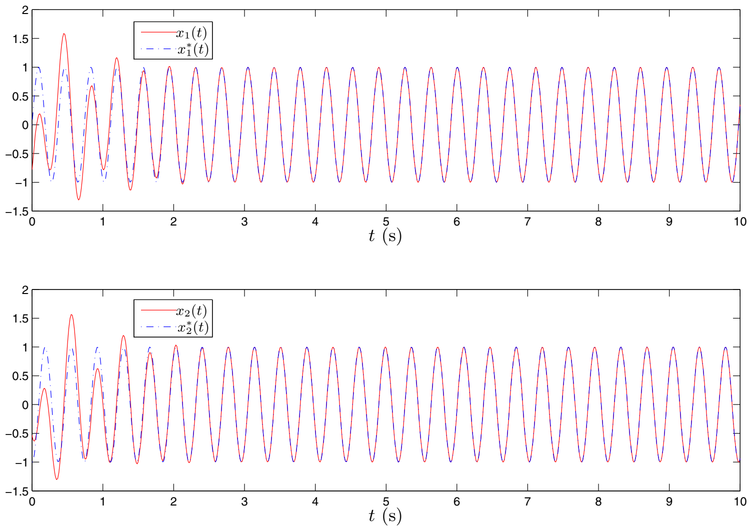


Fig. 1. Trajectories of state of ZNN (4) and theoretical solution of (6)

2.2 GNN Model

For comparison, it is worth pointing out here that we can develop a gradient-based neural network to solve online the quadratic program. However, similar to almost all numerical algorithms and neural-dynamic schemes mentioned before, the gradient neural networks are designed intrinsically for problems with constant coefficient matrices and/or vectors. Now we show the GNN design procedure as the following.

1) Firstly, a scalar-valued norm-based nonnegative energy function, such as $\|Wy+u\|_2^2/2$ with $\|\cdot\|_2$ denoting the two norm of a vector, is constructed such that its minimum point is the solution of linear system $Wy=-u$.

2) Secondly, an algorithm is designed to evolve along a descent direction of this energy function until the minimum point is reached. The typical descent direction is the negative of the gradient of energy function $\|Wy+u\|_2^2/2$, i.e.,

$$-\frac{\partial\|Wy+u\|_2^2/2}{\partial y} = -W^T\big(Wy+u\big).$$

3) Thirdly, by using the above negative gradient to construct and apply the neural network to the time-varying situation, we could have a linear GNN model solving (1),

$$\dot{y}(t) = -\gamma(t)W^T(t)W(t)y(t) - \gamma(t)W^T(t)u(t),$$

and a generalized nonlinear GNN model,

$$\dot{y}(t) = -\gamma(t)W^T(t)\mathcal{F}\big(W(t)y(t)+u(t)\big). \quad (5)$$

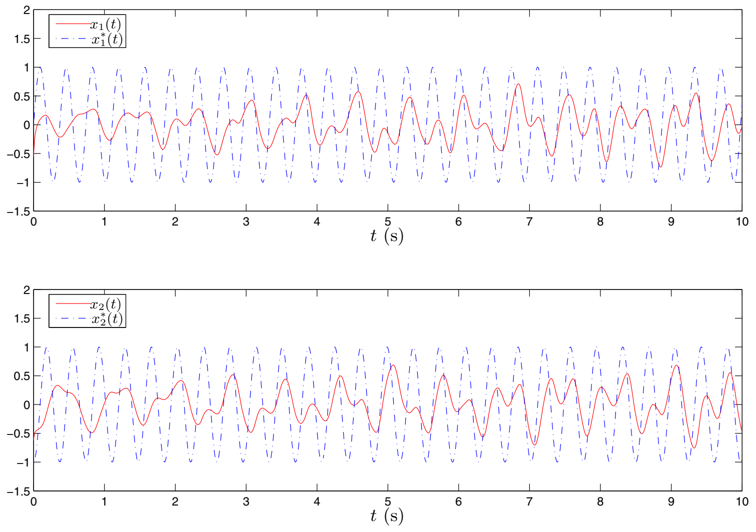


Fig. 2. Trajectories of state of GNN (5) and theoretical solution of (6)

The main differences and novelties of ZNN (4) from GNN (5) may lie in the following facts.

1) The design of ZNN model (4) is based on the elimination of every element of the vector-valued indefinite unbounded error function $e(t) = W(t)y(t) + u(t)$. In contrast, the design of GNN model (5) is based on the elimination of the scalar-valued norm-based nonnegative energy function $\|Wy + u\|_2^2$.

2) ZNN model (4) is depicted in an implicit dynamics, i.e., $W(t)\dot{y}(t) = \cdots$, which coincides well with systems in nature and in practice. In contrast, GNN model (5) is depicted in an explicit dynamics, i.e., $\dot{y}(t) = \cdots$.

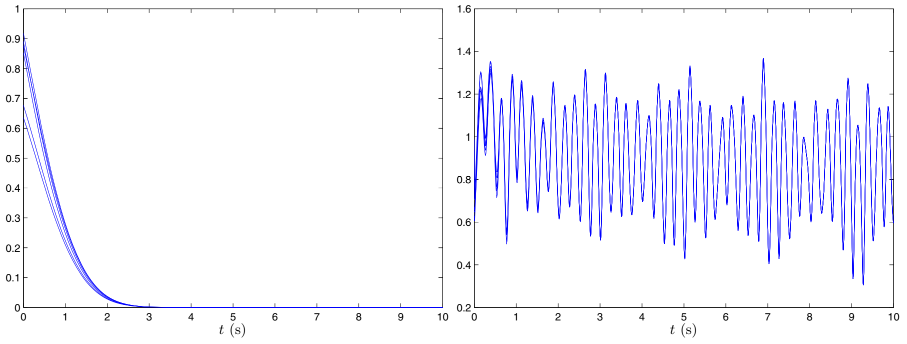
3) ZNN model (4) systematically and methodologically exploits the time-derivative information of coefficient matrices and vectors during its real-time solving process. In contrast, GNN model (5) has not exploited such important information, thus less effective on time-varying problems solving.

3 Simulation Results

In this section, an illustrative example is given to demonstrate the efficiency of ZNN model (4) with the time-varying design parameter $\gamma(t)$ for the online solution of time-varying quadratic programming (1).

Let us consider the following time-varying quadratic program subject to time-varying linear equalities:

$$\begin{aligned}\text{minimize}\quad & \frac{1}{4}(\cos(6t)+2)x_1^2(t) + \frac{1}{4}(\sin(6t)+2)x_2^2(t) + \cos(6t)x_1(t)x_2(t) \\ & + \sin(8t)x_1(t) + \cos(8t)x_2(t),\end{aligned} \tag{6}$$



(a) ZNN model [4] (b) GNN model [5]

Fig. 3. Residual error $\|A(t)x(t) - b(t)\|_2$ of equality-constraint satisfaction by ZNN model [4] and GNN model [5] when solving time-varying quadratic program [6]

$$\text{subject to } \begin{cases} \sin(9t)x_1(t) - \cos(9t)x_2(t) = \cos(8t), \\ \cos(9t)x_1(t) + \sin(9t)x_2(t) = \sin(8t). \end{cases}$$

For purposes of illustration and comparison, both ZNN model [4] and GNN model [5] are exploited to solve online such a time-varying quadratic program [6], which are all equipped with the time-varying design parameter $\gamma(t) = t + 0.5$ and an array of power-sum activation functions with $N = 3$.

From Fig. 1, we observe that state vector $x(t) \in R^2$ of ZNN model [4] elegantly converges to the time-varying theoretical solution $x^*(t) \in R^2$ of time-varying quadratic program [6] within around 2s. However, state vector $x(t) \in R^2$ of GNN model [5] could not converge well to $x^*(t)$, instead with a large error lagged behind the theoretical optimal solution $x^*(t) \in R^2$ of [6], which is illustrated in Fig 2.

As shown in Fig. 3, the residual error $\|A(t)x(t) - b(t)\|_2$ of the time-varying linear equality constraint $A(t)x(t) - b(t) \in R^2$ synthesized by ZNN model [4] for solving time-varying quadratic program [6] diminishes to zero within about 2.7s. However, the residual error $\|A(t)x(t) - b(t)\|_2$ synthesized by GNN model [5] is rather larger with obvious oscillations. This substantiates the efficacy and superiority of ZNN model [4] for solution of [6].

4 Conclusions

In this paper, we have extended the design formula of Zhang neural network (ZNN) to a general one, i.e., the design parameter γ is considered time-varying rather than constant, and investigated the new ZNN model for solving online the time-varying quadratic program subject to time-varying linear-equality constraints. The global exponential convergence and superior convergence of such a new ZNN model have been guaranteed theoretically and substantiated via a computer-simulation example.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grants 61075121, 60935001 and 60775050, and also by the Fundamental Research Funds for the Central Universities of China. In addition, the corresponding author, Y. Zhang, would like to thank and encourage the student author, Z. Li, for his important contribution to the ZNN research, with the following thoughts: "A generation of people might only do things of that generation. As scientific researchers, we have to make progress slow or fast, but preferably steadily and correctly".

References

1. Tank, D.W., Hopfield, J.J.: Simple 'Neural' Optimization Network: An A/D Converter Signal Decision Circuit and a Linear Programming Circuit. IEEE Transactions on Circuits and Systems 33, 533–541 (1986)
2. Wang, J., Zhang, Y.: Recurrent Neural Networks for Real-Time Computation of Inverse Kinematics of Redundant Manipulators. In: Machine Intelligence Quo Vadis?. World Scientific, Singapore (2004)
3. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
4. Liu, J., Brooke, M.A., Hirotsu, K.: A CMOS Feedforward Neural-Network Chip with On-Chip Parallel Learning for Oscillation Cancellation. IEEE Transactions on Neural Networks 13(5), 1178–1186 (2002)
5. Zhang, Y., Wang, J.: A Dual Neural Network for Convex Quadratic Programming Subject to Linear Equality and Inequality Constraints. Physics Letters A 298(4), 271–278 (2002)
6. Zhang, Y., Jiang, D., Wang, J.: A Recurrent Neural Network for Solving Sylvester Equation with Time-Varying Coefficients. IEEE Transactions on Neural Networks 13(5), 1053–1063 (2002)
7. Zhang, Y., Ge, S.S.: Design and Analysis of a General Recurrent Neural Network Model for Time-Varying Matrix Inversion. IEEE Transactions on Neural Networks 16(6), 1477–1490 (2005)
8. Zhang, Y., Li, Z., Yi, C., Chen, K.: Zhang Neural Network versus Gradient Neural Network for Online Time-Varying Quadratic Function Minimization. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 807–814. Springer, Heidelberg (2008)
9. Zhang, Y., Li, Z.: Zhang Neural Network for Online Solution of Time-Varying Convex Quadratic Program Subject to Time-Varying Linear-Equality Constraints. Physics Letters A 373(18-19), 1639–1643 (2009)
10. Li, Z., Zhang, Y.: Improved Zhang Neural Network Model and its Solution of Time-Varying Generalized Linear Matrix Equations. Expert Systems with Applications 37(10), 7213–7218 (2010)
11. Mead, C.: Analog VLSI and Neural Systems. Addison-Wesley, Reading (1989)

Local and Global Burst Synchronization in a Noisy Small-World Neuronal Network

Fang Han[1], Ying Du[2], and Qishao Lu[3]

[1] School of Information Science and Technology, Donghua University, Shanghai 201620, P.R. China
[2] Institute for Cognitive Neurodynamics, School of Information Science and Engineering, East China University of Science and Technology, Shanghai, 200237, P.R. China
[3] Department of Dynamics and Control, Beihang University, Beijing 100191, P.R. China
yadiahan@163.com

Abstract. Chaotic bursting is a fundamental behavior of neurons. In this paper, local and global burst synchronization is studied in a noisy small-world neuronal network composed of nonidentical Hindmarsh-Rose neurons. It is found that burst synchronization can be obtained easily by very small coupling strength and local burst synchronized clusters have already formed before global burst synchronization happens. The effects of the shortcut-adding probability and noise intensity on local and global burst synchronization of the network are also studied and it is found that the introduction of shortcuts facilitates burst synchronization while noise has little effect.

Keywords: burst synchronization; neuronal network; small-world; noise.

1 Introduction

The human brain is a complex network consisting of more than 10^{11} neurons, and each neuron in the cortex connects to more than $10,000$ neurons via synapses [1]. Although the real structure of the neuronal network of human brain is not clear yet, small-world properties [2], such as dense clustering and short average path length, have been found in some neuronal networks [3, 4].

Bursting is a fundamental regime of neuronal behavior. It is believed that bursting has important functions in learning, cognition, motivation, movement control, increasing reliability of cortical synapses and provoking neural disorders [5]. Especially, the synchronous spiking-bursting activity of neurons is thought to be very important for signal transmission and coding in the brain. A lot of studies have been carried out on synchronization in coupled bursting neuronal networks and mainly focused on phase synchronization [6-8] or complete synchronization [9-11]. However, interacting bursting neurons may also show other forms of synchrony, such as burst synchronization, due to the characteristic of multiple time scales, but only a few of researchers studied burst synchronization in neuronal networks.

© Springer-Verlag Berlin Heidelberg 2011

As chaotic bursting is a characteristic of neurons, we aim to study the influence of network topology and noise on local and global burst synchronization in a small-world neuronal network of nonidentical chaotic Hindmarsh-Rose (HR) neurons. This article is organized as follows. Section 2 introduces a model of small-world HR neuronal network. Section 3 presents the main results of burst synchronization. The closing remarks are given in Section 4.

2 Neuronal Network Model

The Hindmarsh-Rose (HR) neuronal model, as a typical example of real neurons, is expressed by the following equations [12]:

$$\begin{aligned} \dot{x} &= y - ax^3 + bx^2 - z + I, \\ \dot{y} &= c - dx^2 - y, \\ \dot{z} &= r[s(x+\chi) - z], \qquad i = 1, 2, \cdots, N \end{aligned} \tag{1}$$

where x is the membrane potential, y is associated with the fast current of the Na+ or K+ ions, and z is associated with the slow current of, for example, the Ca2+ ions. We choose the parameters as $a = 1, b = 3, c = 1, d = 5, s = 4, r = 0.006, \chi = 1.6$. I is the stimulus current which is delivered to the neuron from its external environment and make neurons be in different states.

Neurons in human brain are coupled by electrical synapses or chemical synapses. Using the above neuronal model and Newman-Watts small-world strategy [13], with noise considered, a neuronal network composed of electrically coupled HR neurons can be set up, which is expressed as follows:

$$\begin{aligned} \dot{x_i} &= y_i - ax_i^3 + bx_i^2 - z_i + I_i + \xi_i + \sigma \sum_{j=1}^{N} g_{ij}(x_j - x_i) \ , \\ \dot{y_i} &= c - dx_i^2 - y_i \ , \\ \dot{z_i} &= r[s(x_i+\chi) - z_i], \qquad i = 1, 2, \cdots, N, \end{aligned} \tag{2}$$

where N is the overall number of neurons, σ is the coupling strength, $\mathbf{G} = \{g_{ij}\}_{N\times N}$ is the coupling matrix and ξ_i is the Gaussian white noise applied to neuron i, which satisfies $\langle \xi_i \rangle = 0$ and $\langle \xi_i(t)\xi_j(t')\rangle = D\delta_{i,j}\,\delta(t-t')$, where D represents the noise intensity. In this model, we set $I_i = 3.1 \pm 0.2 \times \text{rand}\,(i = 1, 2, \cdots, N)$ to make all the neurons be in different chaotic bursting states, where *rand* is a random number in the interval $[0, 1]$.

The coupling matrix G is determined by the network structure. Here, the Newman-Watts small-world strategy is adopted, which is expressed as follows: we start with a ring of N neurons, each coupled diffusively to its k nearest neighbors, which means the initial degree of each node of the network is k. Then we add shortcuts between pairs of nodes with probability p, which actually is the connection density of the network. If neuron i is connected with neuron j, then $g_{ij} = 1$, otherwise, $g_{ij} = 0$. As special cases, for $p = 0$, we have the original regular network, and for $p = 1$, we have a globally coupled network. In the following simulations, we fix the number of neurons $N = 100$ and the original node degree $k = 4$.

3 Simulation Results

3.1 Local and Global Burst Synchronization

Chaotic bursting is a multiple-time-scale dynamical behavior. Bursting synchronization means groups of spikes, say bursts, occur simultaneously while the spikes in them occur in different time and amplitudes. Figure 1 shows the state of burst synchronization of two neurons.

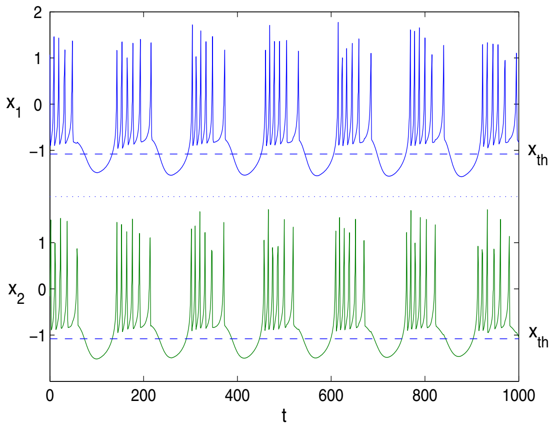


Fig. 1. Burst synchronization and firing threshold of two HR neurons

To study burst synchronization, we choose a threshold value $x_{\text{th}} = -1.08$ for firing occurrence in neurons. Whenever the membrane potential x_i crosses x_{th} in an upwards direction, a burst is thought to happen. Assume that the burst phase ϕ_i is increased by 2π for each burst of the i_{th} neuron and the phase increases linearly between two neighboring bursts. So the burst phase of the i_{th} neuron can be defined as [7]

$$\phi_i(t) = 2\pi k + 2\pi \frac{t - t_k}{t_{k+1} - t_k}, \qquad t_k < t < t_{k+1} \tag{3}$$

where t_k is the time at which the k_{th} burst starts. We define a group of neurons (the number of neurons in a group should be larger than 2) with the same burst phase over time as a cluster of the network, and denote the number of clusters as n_{cluster} and the total number of neurons involved in all clusters as n_{syn}.

Fixing shortcut-adding probability $p = 0.1$ and noise intensity $D = 0.05$, the variation of the number of clusters and that of neurons involved in burst synchronization with respect to the coupling strength can be seen in Figure 2. It can be seen that even for very weak coupling strength, say $\sigma = 0.001$, there are already 10 clusters involving 50 neurons in the network, which indicate the existence of local burst synchronization to a certain extent. With the increasing of the coupling strength, both the number of clusters and the number of synchronized neurons are changing irregularly. However, n_{syn} increases to 100 and n_{cluster} becomes 1 at $\sigma = 0.0012$ and remain constant since then. That means the network achieves global burst synchronization and all the neurons in the network form a whole burst synchronized cluster.

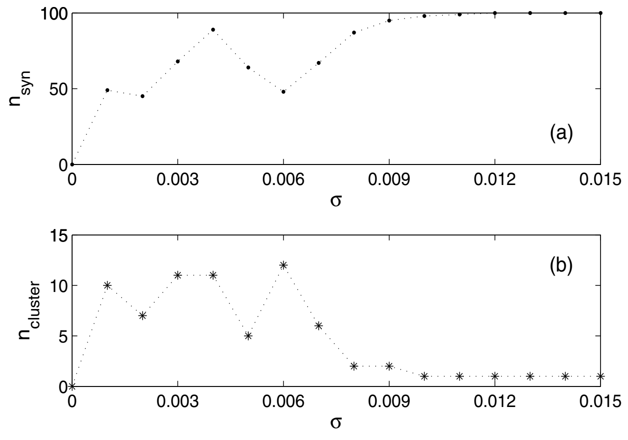


Fig. 2. The variation of the number of synchronized neurons n_{syn} and the number of clusters n_{cluster} with the coupling strength σ

It is believed that the formation of clusters has a great functional significance in the developing process of neuronal networks in human brain because such clusters are rather suitable for information exchanging [14]. Each cluster may have different frequencies and then can be used to transmit information in a particular bandwidth, which may provide a multi-channel processing of information. They also provide a multi-channel communication so that information can arrive simultaneously in different places of the network. We can also suppose that all these clusters can be used to carry different information. That may be why our brain, which is composed of billions of neurons, can accommodate so much information.

The number of clusters n_{cluster} and the number of synchronized neurons n_{syn} both change irregularly with varying coupling strength σ, so both the two parameters can not describe visually the extent of burst synchronization. Thus, we introduce the average cluster volume, which is defined as:

$$\langle n \rangle = \frac{n_{\text{syn}}}{n_{\text{cluster}}} \tag{4}$$

Here, $\langle n \rangle$ is an average number of burst synchronized neurons per cluster of the network. The larger $\langle n \rangle$ is, the more burst synchronized the network is. When $\langle n \rangle = N$ ($N = 100$ in this article), global burst synchronization is achieved.

We plot the variation of $\langle n \rangle$ with the coupling strength σ in Figure 3. It can be seen that the network becomes more and more synchronous in bursts when the coupling strength is increased and achieves global burst synchronization finally for $\sigma \geqslant 0.012$.

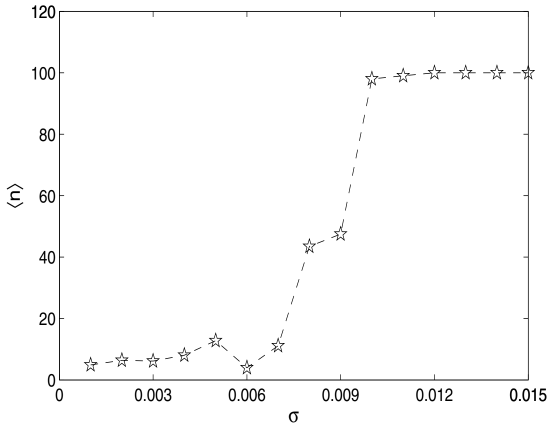


Fig. 3. The relationship of the average cluster volume $\langle n \rangle$ and the coupling strength σ

3.2 Effect of the Shortcut-Adding Probability

We set the parameters $\sigma = 0.001$, $D = 0.05$ and study the effect of the shortcut-adding probability p on burst synchronization. The result is shown in Figure 4. Clearly, global burst synchronization cannot be achieved in the network at $\sigma = 0.001$ no matter how large p is. However, introduction of shortcuts to the neuronal network facilitates local burst synchronization. Generally, the more shortcuts the network has, the more synchronous in bursts the network is. That's probably because the introduction of shortcuts decreases the average path-length of the network and make information transmission more smoothly.

We also calculate the critical coupling strength σ_c for global burst synchronization when varying shortcut-adding probability p, with noise intensity fixed $D = 0.05$. The result is shown in Figure 5. It can be seen that the critical coupling strength for global burst synchronization decreases with increasing p, which means more shortcuts help improving synchronization of the network. As mentioned by many literatures, the regular network ($p = 0$) is very difficult to be synchronized and it is found here that the critical coupling strength σ_c for $p = 0$ is almost up to 9. Generally speaking, the critical coupling strengths for global burst synchronization are considerably small for a variety of network topologies so global burst synchronization can be easily obtained in neuronal networks. The result here is an indication that burst synchronization may have a greater significance than other forms of synchronization because it can be much more easily obtained than other synchronization states.

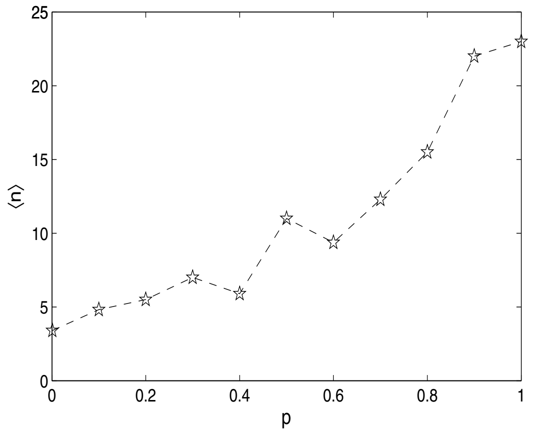


Fig. 4. The relationship of the average cluster volume $\langle n \rangle$ and the adding probability p

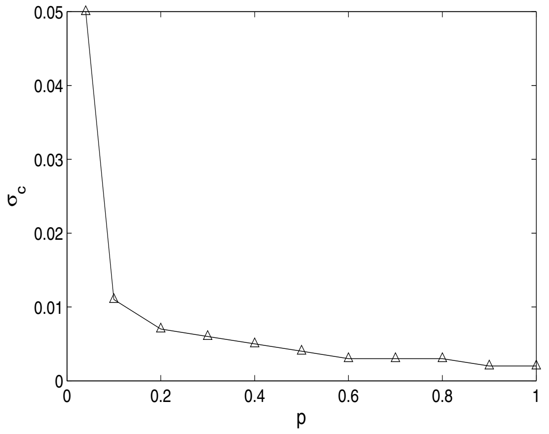


Fig. 5. The relationship of the critical coupling strength of global burst synchronization σ_c and the adding probability p

3.3 Effect of the Noise

Considering neurons always locate in noisy environment, we add Gaussian white noise in the neuronal network model. Fixing $p = 0.1$, the variations of the number of clusters n_{cluster}, the number of synchronized neurons n_{syn} and the average cluster volume $\langle n \rangle$ with different noise intensity D are shown in Figure 6.

It is shown in Figure 6 that n_{cluster}, n_{syn} and $\langle n \rangle$ all change irregularly but not much when noise D intensity increases from 0 to 0.5. That is to say, noise has little effect on burst synchronization of neuronal networks. In other words, burst synchronization is robust to noise. This is another indication that burst synchronization has a greater significance for intercommunication of neurons.

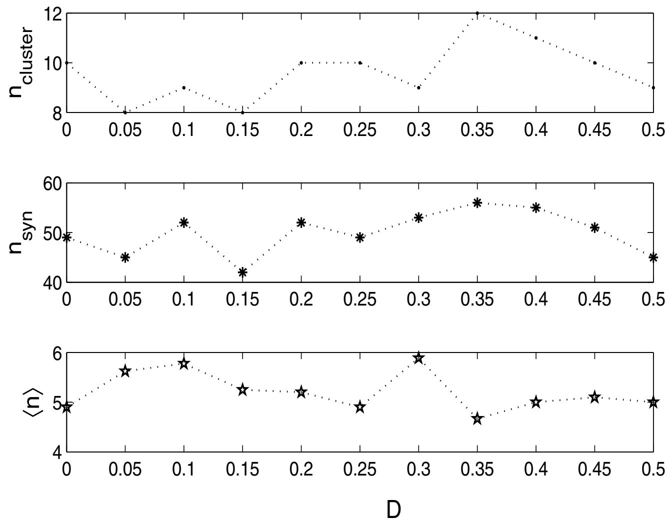


Fig. 6. The variations of n_{cluster}, n_{syn} and $\langle n \rangle$ and noise intensity D

4 Closing Remarks

In this article, we study burst synchronization of a small-world neuronal network of nonidentical HR neurons, with noise considered. First we study the onset of local and global burst synchronization. It is found that the burst synchronization is easily obtained and some local clusters can be formed prior the network achieves the global burst synchronization. Then we investigate the effect of shortcut-adding probability and noise intensity on burst synchronization of the network. The results show that the introduction of shortcuts improves burst synchronization but noise has little effect on it. The above results strongly indicate the great significance of burst synchronization in the information activities of neuronal networks.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (Nos.61075105, 11002103 and 11002055), State Key Laboratory of Industrial Control Technology (No.ICT1007), Shanghai Education Development Foundation Chenguang Project (No.10CG33), the Major Program for the Fundamental Research of Shanghai (No.09JC1400700) and the Fundamental Research Funds for the Central Universities.

References

1. Gerstner, W., Kistler, W.M.: Spiking neuron models. Cambridge University Press, Cambridge (2002)
2. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)

3. Sporns, O., Tononi, G., Edelman, G.M.: Theoretical neuroanatomy: relating anatomical and functional connectivity in graphs and cortical connection matrices. Cereb Cortex 10, 127–141 (2000)
4. Stephan, K.E., et al.: Computational analysis of functional connectivity between areas of primate cerebral cortex. Philos. Trans. R. Soc. Lond. B 355, 111–126 (2000)
5. Ivanchenko, M.V., Osipov, G.V., Shalfeev, V.D., Kurths, J.: Network mechanism for burst generation. Phys. Rev. Lett. 108101 (2007)
6. Shuai, J.W., Durand, D.M.: Phase synchronization in two coupled chaotic neurons. Phys. Lett. A 264, 289–297 (1999)
7. Ivanchenko, M.V., Osipov, G.V., Shalfeev, V.D., Krths, J.: Phase synchronization in ensembles of bursting oscillators. Phys. Rev. Lett. 93, 134101 (2004)
8. Wang, Q.Y., Lu, Q.S.: Phase synchronization in small-world chaotic neural networks. Chin. Phys. Lett. 22, 1329–1332 (2005)
9. Wang, Q.Y., Lu, Q.S., Chen, G.R., Guo, D.H.: Chaos synchronization of coupled neurons with gap junctions. Phys. Lett. A 356, 17–25 (2006)
10. Wang, H.X., Lu, Q.S., Wang, Q.Y.: Complete Synchronization in Coupled Chaotic HR Neurons with Symmetric Coupling Schemes. Chin. Phys. Lett. 22, 2173–2175 (2005)
11. Yoshioka, M.: Chaos synchronization in gap-junction-coupled neurons. Phys. Rev. E 71, 065203(R) (2005)
12. Hindmarsh, J.L., Rose, R.M.: A model of the nerve impulse using two first-order differential equations. Nature 296, 162–164 (1982)
13. Newman, M.E.J., Watts, D.J.: Renormalization group analysis of the small-world network model. Phys. Lett. A 263, 341–346 (1999)
14. Pereira, T., Baptista, M.S., Kurths, J.: General framework for phase synchronization through localized sets. Phys. Rev. E 75, 026216 (2007)

H_∞ Synchronization Control in Nonlinear Time-Delay Complex Dynamical Network

Ting Xiang and Minghui Jiang

Institute of Nonlinear Complex Systems, College of Science,
China Three Gorges University, Yichang, Hubei, 443002, China
xiangting0123@163.com

Abstract. On the basis of Lyapunov stability theory and LMI technique, this paper investigates H_∞ synchronization control of time-varying synchronization state in general complex networks with time-delay and external disturbances. H_∞ synchronization controllers have been designed for nodes of controlled network and the H_∞ control law for asymptotically(exponentially) synchronization are derived via defining an appropriate controlled output. Besides, under the desired H_∞ performance index, the control criteria which guarantee complex networks synchronization with a time-varying state are revealed by LMI. Finally, a simulation example is given to demonstrate the effectiveness of the theoretical results.

Keywords: Complex networks; time-delay; H_∞ control; Asymptotically synchronization; Exponentially synchronization; LMI.

1 Introduction

Synchronization is widely regarded as a kind of collective behavior which exists in many dynamic systems. Because of its importance in various practical applications, more and more researchers are working in this field. Among these endeavors, Wang and Chen [1,2] analyzed the synchronization based on simple dynamical network model. Then Li and Chen [3] extended the model to the one with coupling delays. Lv [4] analyzed the synchronization with periodic orbits of a time-varying dynamical network model. However, it is noticed that many complex networks in practice are very difficult to synchronize without man-made controller. Thus, synchronization control is introduced to tolerate the failures of complex systems synchronization and performance. While it is significant to design an appropriate controller that ensures the system can reach the desired orbit in synchronization control problem.

There are also authors having investigated adaptive synchronization control [5] and pinning synchronization control [6]. However, most of the research in general complex networks assumed the model was in an ideal condition without the external disturbance, which is always existent and might cause the complex network to diverge or oscillate. So it is imperative to enhance the anti-interference ability of the system. However, to my knowledge, fewer works study the H_∞ synchronization control [7,8]

© Springer-Verlag Berlin Heidelberg 2011

in complex networks, whose performance index can precisely reflect the ability of anti-interference. So, we will analyze the H_∞ synchronization control problem of coupling time-delay [9] nonlinear general complex system and obtain related conclusion via designing appropriate controller.

Recently, synchronization problem in general complex networks with time-delay is widely studied by using LMIs technique. We can also transform the H_∞ synchronization control problem into solving LMI by constructing Lyapunov function. In other words, we only need to solve a LMI to obtain appropriate controller and control law which could guarantee the synchronism of the complex networks.

This paper is organized as follows: Section 2 introduces a general complex network with time-delay coupling and obtains the error equation under feedback control; In Section 3, based on Lyapunov function and LMI, we consider the problem of H_∞ synchronization control of nonlinear time-delay complex systems. While the robust synchronization H_∞ control criterion for the nonlinear time-delay complex system is obtained. Section 4 gives a simple example to show the effectiveness of the proposed synchronization control criteria.

2 Problem Formulation and Preliminaries

We consider a general complex network consisting of N dynamical nodes. Each node of the network is a $n-$dimensional autonomous dynamical system with time-delay, which is described by:

$$\dot{x}_i(t) = f(x_i(t)) + c\sum_{j=1}^{N} a_{ij} H x_j(t-\tau) + \omega_i(t) + u_i(t) \qquad i = 1,\cdots,N \tag{1}$$

where $x(t) = (x_{i1}(t),\cdots,x_{in}(t))^T \in R^n$ are state variables of the i th dynamical node; the constant $c > 0$ is coupling strength; $A = (a_{ij})_{N\times N} \in R^{N\times N}$ is outer-coupling matrix, in which a_{ij} is defined as follows: if exist connection between the nodes i and $j(j \neq i)$, then $a_{ij} = a_{ji} = 1$; otherwise, $a_{ij} = a_{ji} = 0$, and the diagonal elements of matrix A are defined by $a_{ii} = \sum_{j=1, j\neq i}^{N} a_{ij}, i = 1,2,\cdots,N$, while A is irreducible. $H \in R^{n\times n}$ is a constant inner-coupling matrix. $f(x_i): R^n \times R^n \to R^n$ represents the state of every single node, which is continuously, differentiable and exists a unique continuous solution for any initial condition (t_0, x_{i0}), where x_{i0} is an $n-$ dimensional vector.

Let $s(t)$ be a solution of the controlled system (1), and we say controller u_i asymptotically (exponentially) solves the synchronization problem with time-varying reference state, if $\lim_{t\to\infty}\|x_i(t) - s(t)\| = 0 \quad ,i = 1,\cdots,N$, where $\dot{s}(t) = f(s(t))$. So the synchronization state is $S(t) = (s^T(t),\cdots,s^T(t))^T$. To solve this problem, we use the following synchronization controller:

$$u_i(t) = -k_i H(x_i(t) - s(t)), i = 1, 2, \cdots, N \tag{2}$$

Define error vectors as: $\Delta_i(t) = x_i(t) - s(t), i = 1, 2, \cdots .N$.

According to the controlled system (1), the error system is then described by:

$$\dot{\Delta}_i(t) = \dot{x}_i(t) - \dot{s}(t) = f(x_i(t)) - f(s(t)) + c\sum_{j=1}^{N} a_{ij} H\Delta_j(t-\tau) - k_i H\Delta_i(t) + \omega_i(t) \tag{3}$$

Where $1 \le i \le N$. Next, we do linear transformation at the equilibrium point $s(t) \in R^n$:

$$\dot{\Delta}_i(t) = Df(s(t))\Delta_i(t) + c\sum_{j=1}^{N} a_{ij} H\Delta_j(t-\tau) - k_i H\Delta_i(t) + \omega_i(t) \tag{4}$$

where $1 \le i \le N$, and $Df(s(t))$ is the Jacobean matrix at the point of $s(t)$. Let $J_i = \frac{\partial f}{\partial s}$, so (3) can be written in a matrix form:

$$\dot{\Delta}(t) = (A - KH)\Delta(t) + B\Delta(t-\tau) + \omega(t) \tag{5}$$

Where $A = \begin{bmatrix} J_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & J_N \end{bmatrix}$, $B = \begin{bmatrix} ca_{11}H & \cdots & ca_{1N}H \\ \vdots & \ddots & \vdots \\ ca_{N1}H & \cdots & ca_{NN}H \end{bmatrix}$, $K = \begin{bmatrix} k_1 I_n & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & k_N I_n \end{bmatrix}$.

Obviously, the synchronization performance of original system is equivalent to the stability of system (5).

Meanwhile, in order to let synchronization state suppress external disturbances and make all subsystem reach agreement, we define the output function of system as follow:

$$z(t) = U\Delta(t), \Delta(0) = 0 \tag{6}$$

Then the dynamic system can be summarized as follow:

$$\begin{aligned} &\dot{\Delta}(t) = (A - KH)\Delta(t) + B\Delta(t-\tau) + \omega(t) \\ &z(t) = U\Delta(t), \Delta(0) = 0 \end{aligned} \tag{7}$$

The ability of complex system against external disturbances can be measured by the norm of the closed-loop transfer function $T_{\omega s}(s)$ which is expressed by:

$$\|T_{\omega s}(s)\|_\infty = \sup_{0 \ne \omega(t) \in l_2[0,\infty)} \frac{\|z(t)\|_2}{\|\omega(t)\|_2} \tag{8}$$

Here, we can make the closed-loop system with the given H_∞ performance index $\gamma > 0$ satisfy: $\|T_{\omega s}(s)\|_\infty < \gamma$, and it can be described by:

$$J(\omega) = \int_0^\infty \left(z^T(t)z(t) - \gamma^2 \omega^T(t)\omega(t) \right) dt < 0 \tag{9}$$

Hence, our goal is to design controller $u(t)$, and for any given positive definite constant $\gamma > 0$, the closed loop system satisfied: (a) The closed loop system is asymptotically (exponentially) synchronization, when $\omega(t)=0$; (b) Under zero-value initial condition, $J(\omega)=\int_0^\infty \left(z^T(t)z(t)-\gamma^2\omega^T(t)\omega(t)\right)dt<0$.

3 Main Results

In this section, we will provide the synchronization control analysis of the nonlinear general complex dynamical network with coupling time-delay and external disturbances. And the sufficient conditions will be presented by LMI for general complex dynamical network.

Definition: For a given $\gamma>0$ and $\alpha>0$, selecting control law, asymptotically (exponentially) solves the synchronization problem with time-varying reference state under $\|T_{\omega s}(s)\|_\infty<\gamma$ is: (1) The closed loop system (7) is α asymptotically (exponentially) synchronization when $\omega(t)=0$; (2) Under zero-valued initial condition, the closed loop system (7) satisfied: $J(\omega)=\int_0^\infty \left(z^T(t)z(t)-\gamma^2\omega^T(t)\omega(t)\right)dt<0$ for all nonzero $\omega(t)\in L_2[0,+\infty)$ [9].

3.1 Asymptotically Synchronization

Theorem 1. Consider a nonlinear general complex dynamical network (1) with coupling time-delay τ, for a given index $\gamma>0$, controller (2) asymptotically solves the synchronization problem with time-varying reference state under $\|T_{\omega s}(s)\|_\infty<\gamma$, if for the closed loop system (7), there exists positive definite symmetric matrices $P,Q\in R^{Nn\times Nn}$ and real matrix W, satisfying:

$$\Omega=\begin{bmatrix} W+W^T+Q & PB & P & U^T \\ B^TP & -Q & 0 & 0 \\ P & 0 & -\gamma^2 I & 0 \\ U & 0 & 0 & -I \end{bmatrix}<0 \tag{10}$$

Where $W=P(A-KH)$.

Proof: (1) Firstly, we need to prove the asymptotically stable under zero-valued initial condition, that is $\omega(t)=0$. Constructing the Lyapunov function as follows:

$$V=\Delta^T(t)P\Delta(t)+\int_{t-\tau}^{t}\Delta^T(s)Q\Delta(s)ds$$

Where $P,Q\in R^{Nn\times Nn}$ are positive define symmetric matrices. Along the trajectory of system (7), let $\xi^T(t)=\left[\Delta^T(t)\quad \Delta^T(t-\tau)\right]$, then we have:

$$\begin{aligned}\dot V &= \dot\Delta^T(t)P\Delta(t)+\Delta^T(t)P\dot\Delta(t)+\Delta^T(t)Q\Delta(t)-\Delta^T(t-\tau)Q\Delta(t-\tau)\\ &=\xi^T(t)\begin{bmatrix}(A-KH)^TP+P(A-KH)+Q & PB\\ B^TP & -Q\end{bmatrix}\xi(t)\end{aligned}$$

Under the given conditions by the **Theorem1**, $\dot V\le 0$. According to definition1, the original system (1) is asymptotically synchronization;

(2)For any nonzero $\omega(t)\in L_2[0,+\infty)$, $t>0$ and zero-valued initial condition, that $\Delta(s)=0, s\in(-\infty,0)$, let $\eta^T(t)=\begin{bmatrix}\Delta^T(t) & \Delta^T(t-\tau) & \omega^T(t)\end{bmatrix}$ we have:

$$\begin{aligned}
J_{\omega z} &= \int_0^\infty (z^T(t)z(t)-\gamma^2\omega^T(t)\omega(t))dt \\
&= \int_0^\infty (\Delta^T(t)U^TU\Delta(t)-\gamma^2\omega^T(t)\omega(t)+\dot{V})dt - V(t)+V(0) \\
&\le \int_0^\infty (\Delta^T(t)U^TU\Delta(t)-\gamma^2\omega^T(t)\omega(t)+\dot{V})dt \\
&= \int_0^\infty \eta^T(t)\begin{bmatrix}(A-KH)^TP+P(A-KH)+Q+U^TU & PB & P \\ B^TP & -Q & 0 \\ P & 0 & -\gamma^2 I\end{bmatrix}\eta(t)dt
\end{aligned}$$

Let $P(A-KH)=W$, while according to $\Omega<0$, we can conclude that $J_{\omega z}\le 0$. And the control law K can be obtained by calculating LMI (10).

Therefore, asymptotically synchronization with time-varying reference state under $\|T_{\omega s}(s)\|_\infty<\gamma$ can be achieved if the matrix inequality (10) holds.

Remark 1: The robust H_∞ asymptotically synchronization controller design problem is solved in **Theorem 1** for the addressed nonlinear complex network, which was seldom discussed in the past literature. We transform the H_∞ synchronization control problem into solving LMI, which make it very easy to operate in practical systems.

3.2 Exponentially Synchronization

Theorem 2. Consider a nonlinear general complex dynamical network (1) with coupling time-delay τ, for a given index $\gamma>0$, controller (2) exponentially solves the synchronization problem with time-varying reference state under $\|T_{\omega s}(s)\|_\infty<\gamma$, if there exists positive definite symmetric matrices $P,Q\in R^{Nn\times Nn}$ and real matrice W, satisfying:

$$\begin{bmatrix}2P+W^T+\alpha P+W+Q & e^{\alpha\tau}PB \\ e^{\alpha\tau}B^TP & -Q\end{bmatrix}<0 \tag{11}$$

$$\begin{bmatrix}W^T+W+Q+U^TU & PB & P \\ B^TP & -Q & 0 \\ P & 0 & -\gamma^2 I\end{bmatrix}<0 \tag{12}$$

Where $W=P(A-KH)$.

Proof: (1) Firstly, we need to prove the exponentially synchronization under zero-valued initial condition, that is $\omega(t)=0$. Here we choose an appropriate transformation $\eta(t)=e^{\alpha t}\Delta(t)$, then:

$$\dot{\eta}(t)=\alpha e^{\alpha t}\Delta(t)+e^{\alpha t}\dot{\Delta}(t)=\alpha\eta(t)+(A-KH)\eta(t)+e^{\alpha\tau}B\eta(t-\tau) \tag{13}$$

It is obvious that the exponentially stable of system (7) is equivalent to the asymptotically stable of this system [10]. So we should check the asymptotically stable of system (13). Constructing the Lyapunov function as follows:

$$V(t) = \eta^T(t)P\eta(t) + \int_{t-\tau}^{t} \eta^T(s)Q\eta(s)ds$$

Where $P, Q \in R^{Nn \times Nn}$ are positive define symmetric matrices. Along the trajectory of system (13), and denote the vector as $\xi^T(t) = \begin{bmatrix} \eta^T(t) & \eta^T(t-\tau) \end{bmatrix}$, then we have:

$$\begin{aligned} \dot{V}(t) &= \dot{\eta}^T(t)P\eta(t) + \eta^T(t)P\dot{\eta}(t) + \eta^T(t)Q\eta(t) - \eta^T(t-\tau)Q\eta(t-\tau) \\ &= \xi^T(t)\begin{bmatrix} 2P + W^T + \alpha P + W + Q & e^{\alpha\tau}PB \\ e^{\alpha\tau}B^T P & -Q \end{bmatrix}\xi(t) \end{aligned}$$

Hence, $\dot{V}(t) \le 0$. The original system (1) is exponentially synchronization.

(2) For any nonzero $\omega(t) \in l_2[0, +\infty)$. Let $\eta^T(t) = \begin{bmatrix} \Delta^T(t) & \Delta^T(t-\tau) & \omega^T(t) \end{bmatrix}$ we have:

$$\begin{aligned} J_{\omega z} &= \int_0^\infty (z^T(t)z(t) - \gamma^2\omega^T(t)\omega(t))dt \\ &= \int_0^\infty (\Delta^T(t)U^T U\Delta(t) - \gamma^2\omega^T(t)\omega(t) + \dot{V})dt - V(t) + V(0) \\ &\le \int_0^\infty (\Delta^T(t)U^T U\Delta(t) - \gamma^2\omega^T(t)\omega(t) + \dot{V})dt \\ &= \int_0^\infty \eta^T(t)\begin{bmatrix} (A-KH)^T P + P(A-KH) + Q + U^T U & PB & P \\ B^T P & -Q & 0 \\ P & 0 & -\gamma^2 I \end{bmatrix}\eta(t)dt \end{aligned}$$

Let $P(A-KH) = W$, and according to (11), we can conclude that $J_{\omega z} \le 0$. And the control law K can be obtained by calculating LMI.

Therefore, exponentially synchronization with time-varying reference state under $\|T_{\omega s}(s)\|_\infty < \gamma$ can be achieved if the matrix inequality (11) and (12) holds.

Remark 2: From the conditions of the theorem, it can be seen that the conditions of **Theorem 1** is delay-independent, while **Theorem 2** is delay-dependent and the upper bound of the delay can be obtained by using the LMI toolbox in MATLAB. So the conditions of **Theorem 2** are less conservative than those of **Theorem 1.**

3.3 System with Time-Varying Delays

To further reduce the conservative, we consider the time-varying delay $\tau(t)$, which satisfies the assumption that:

$$0 \le \tau(t) \le d, \dot{\tau}(t) \le b < 1; \tag{14}$$

Where d and b are constants.

Theorem 3. Consider a nonlinear general complex dynamical network (1) with coupling time-delay $\tau(t)$ which satisfies (14), for a given index $\gamma > 0$, controller (2) exponentially solves the synchronization problem with time-varying reference state under $\|T_{\omega s}(s)\|_\infty < \gamma$, if there exists positive definite symmetric matrices $P, Q \in R^{Nn\times Nn}$ and real mat rice W, satisfying:

$$\begin{bmatrix} 2P + W^T + \alpha P + W + Q & e^{\alpha\frac{d}{1-b}}PB \\ e^{\alpha\frac{d}{1-b}}B^T P & -Q \end{bmatrix} < 0 \tag{15}$$

$$\begin{bmatrix} W^T + W + Q + U^T U & PB & P \\ B^T P & -Q & 0 \\ P & 0 & -\gamma^2 I \end{bmatrix} < 0 \tag{16}$$

Where $W = P(A - KH)$.

The proof of this theorem is similar to that of theorem 2. So it is omitted here.

4 Simulation

In this section, simulation example will be given to verify the validity of the theoretical results obtained in the previous section. For simplicity, we consider a complex network of 4 nodes, and each node represents one-dimensional subsystem.

The state equation can be expressed as:

$$\dot{x}_i(t) = f(x_i(t)) + c\sum_{j=1}^{N} a_{ij} H x_j(t-\tau) + \omega_i(t) + u_i(t) \qquad i = 1, \cdots, N$$

Where $N = 4$. By transforming, the error corresponding system can be obtained:

$\dot{\Delta}(t) = (A - KH)\Delta(t) + B\Delta(t-\tau) + \omega(t)$, where $\omega(t) = [\sin t \quad 0.5\cos 2t \quad \sin t \quad 0.8\cos 2t]$.

$$A = \begin{bmatrix} 0.2 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0.2 \end{bmatrix} \qquad B = \begin{bmatrix} -3 & 1 & 1 & 1 \\ 1 & -3 & 1 & 1 \\ 1 & 1 & -3 & 1 \\ 1 & 1 & 1 & -3 \end{bmatrix}$$

Let the performance index $\gamma = 1$, and the output matrix $U = 0.4I$.

Applying **theorem 1**, we can gain the control law $k_i = 5.2$ $(i = 1, 2, \cdots, N)$ and let $\tau = 0.3s$. The error system with $\|T_{\omega s}(s)\|_\infty < \gamma$ asymptotically converges to $s(t) = 0$ and $s(t) = 5\sin t$. Besides, the performance index $\gamma = 0.9523$ can be obtained, which is less than the pervious chosen performance index $\gamma = 1$.

Applying **theorem 2**, we can gain the control law $k_i = 11.2$ $(i = 1, 2, \cdots, N)$ and let $\tau = 0.3s$. The error system with $\|T_{\omega s}(s)\|_\infty < \gamma$ exponentially converges to $s(t) = 0$; and $s(t) = 5\sin t$. Besides, the performance index $\gamma = 0.9464$ can be obtained, which is less than the pervious chosen performance index $\gamma = 1$.

Without controller, the performance index is γ=1.6178.

Remark 3: By comparing, we summarize that the complex network with controller have stronger anti-interference ability.

5 Conclusion

This paper addresses H_∞ synchronization control with time-varying synchronization state in general complex networks subject to external disturbances. By feedback controller, the Nonlinear general complex dynamical network achieve asymptotically (exponentially) synchronization and satisfied the given performance index, which can be solved by LMIs. This is very significant for practical application. Further, on basis of this article, we can obtain the optimal H_∞ controller and the sufficient conditions in terms of LMIs are given to guarantee the system can reach the time-varying synchronization state under $\|T_{\omega s}(s)\|_\infty < \gamma$. In addition, simulation example is given to illustrate the effectiveness of the conclusions.

Acknowledgments

The work was supported by National Science Foundation of China (No. 61074091), National Science Foundation of Hubei Provincial (2010CDB10807), the Scientific Innovation Team Project of Hubei Provincial Department of Education (T200809) and the Science Foundation of Education Commission of Hubei Province (D20091305, D20101202).

References

1. Wang, X., Chen, G.: Synchronization in Scale-free Dynamical Networks: Robustness and Fragility. IEEE Trans. Circuits and System-I 49(1), 54–62 (2002)
2. Wang, X., Chen, G.: Synchronization in Small-Word Dynamical Networks. Int. J. Bifurcation 12(1), 1987–1992 (2002)
3. Li, C., Chen, G.: Synchronization in General Complex Dynamical Networks with Coupling Delays. Physica A 343, 178–236 (2004)
4. Lv, J., Chen, G.: Time-Varying Complex Dynamical Network Model and Its Controlled Synchronization Criteria. IEEE Trans. Autom. Control 50(3), 841–846 (2005)
5. Zhong, W.-S., Dimirovski, G.M.: Adaptive Synchronization for a Class of Delayed Dynamical Networks. In: American Control Conference, pp. 5121–5126 (2008)
6. Wang, X.F., Chen, G.R.: Pinning Control of Scale-Free Dynamical Networks. Phys. A 310, 521–531 (2002)
7. Khargonekar, P.P., Petersen, I.R., Zhou, K.: Robust Stabilization of Uncertain Linear Systems: Quadratic Stabilizability and H_∞ Control Theory. IEEE Trans. on Automatic Control 35, 356–361 (1990)
8. Li, Y., Jian, C.: an LMI Approach to Guaranteed Cost Control of Linear Uncertain Time-Delay Systems. Automatica 35(6), 1155–1159 (1999)
9. Li, Y., Furong, G.: Robust H_∞ Control of Discrete-Time Linear Systems with Delayed State and Frobenius Norm-Bounded Uncertainties. In: Proceeding of the 39th IEEE Conference on Decision and Control, pp. 1754–2755 (2000)
10. Liu, T., Georgi, M.D., Zhao, J.: Exponential Synchronization of Complex Delayed Dynamical Networks with General Typology. Phys. A 387, 643–652 (2008)

Anti-synchronization and Control of New Chen's Hyperchaotic Systems*

Zunshui Cheng[1,2]

[1] School of Mathematics and Physics, Qingdao University of Science and Technology
Qingdao 266061, China
[2] Department of Mathematics, Southeast University
Nanjing 210096, China
chengzunshui@gmail.com

Abstract. In this paper, a new Chen hyperchaotic system was introduced and analyzed. The anti-synchronization between two different hyperchaotic systems is investigated based on the nonlinear control theory. The sufficient condition is drawn for the stability of the error dynamics, where the controllers are designed by using the sum of the relevant variables in hyperchaotic systems. Numerical simulations are performed to demonstrate the effectiveness of the proposed control strategy.

Keywords: Anti-synchronization; Feedback control; Lyapunov function; Numerical simulation.

1 Introduction

Motivated by the study of chaos synchronization since the pioneering work by Pecora and Carrol [1], an increasing interest has been devoted to study synchronization of neural networks. The aim of synchronization is to use the output of a master system to control the slave system so that its output of the slave system follows that of the master system asymptotically. Synchronization of neural networks have many applications in secure communication and so on. Therefore, the study of neural synchronization is an important step for both understanding brain science and designing neural networks for practical use [2-5].

There are different types of synchronization in interacting nodes of chaotic neural networks, such as complete synchronization, lag synchronization , generalized synchronization, phase synchronization, projective synchronization, generalized projective synchronization, anti-synchronization and other types of synchronization [6-13].

Hyperchaotic system is usually defined as a chaotic system with more than one positive Lyapunov exponent. As we know now, there are many hyperchaotic systems discovered in the high-dimensional social and economical networks. Typical examples

* This work was supported by China Postdoctoral Science Foundation funded project 20100481088, the Natural Science Foundation of Shandong Province of China under Grant ZR2010FL016 and National Natural Science Foundation of China under Grant 11001144.

© Springer-Verlag Berlin Heidelberg 2011

are four-dimensional (4D) hyperchaotic Rösser system, 4D hyperchaotic Lorenz-Haken system, 4D hyperchaotic Chua's circuit, and 4D hyperchaotic Chen's system [14-18]. Since hyperchaotic system has the characteristics of high capacity, high security and high efficiency, it has broadly applied potential in secure communications, neural networks, and so on.

More recently, anti-synchronization of coupled hyperchaotic dynamical systems has received a great deal of attention in many fields of science and technology[9-10]. In which the state vectors of synchronized systems have the same absolute values but opposite signs. Therefore, the sum of two signals can converge to zero when anti-synchronization appears.

In this paper, we will develop the state observer method for constructing anti-synchronized slave system.

The remaining of this paper is organized as follows: In Section 2, we give Lyapunov stability criteria for globally exponential hyperchaos (lag) synchronization of n-dimensional chaotic systems. By using the obtained theory, different feedback controller are investigated for globally exponential (lag) synchronization of hyperchaotic networks. Numerical simulation results are presented to illustrate the analytical predictions in Section 3. Finally, the conclusion is given in Section 4.

2 Preliminaries

Consider the drive chaotic system in the form of[9]

$$\dot{x} = Ax + Bf(x) \tag{1}$$

where $x \in R^n$ is the state vector, $A \in R^{n\times n}$, $B \in R^{n\times n}$ are metrices and vectors of system parameters, and $f : R^n \to R^n$ is a nonlinear function. Eq. (1) is considered as a drive system.

By introducing an additive control $U \in R^n$, then the controlled response system is given by

$$\dot{y} = A_1 y + B_1 g(y) + U \tag{2}$$

where $y \in R^n$ is the state vector, $A_1 \in R^{n\times n}$, $B_1 \in R^{n\times n}$ are metrices and vectors of system parameters, and $g : R^n \to R^n$ is a nonlinear function. Eq. (2) is considered as a slave system. The anti-synchronization problem is to design a controller U which anti-synchronizes the states of both the drive and response systems.

Let $e = y + x$ is the anti-synchronization error vector. Our goal is to design controller u such that the trajectory of the response system (2) with initial condition y_0 can asymptotically approaches the drive system (1) with initial condition x_0 and finally implement the anti-synchronization, in the sense that,

$$lim_{t\to\infty}\|e\| = lim_{t\to\infty}\|y(t,y_0) + x(t,x_0)\| = 0 \tag{3}$$

where $\|\cdot\|$ is the Euclidean norm.

3 Anti-synchronization of Identical Hyperchaotic System

In this section to study the anti-synchronization problem of identical new Chen hyperchaotic system with four state variables. However, the initial condition on the drive system is different from that of the response system. The two hyperchaotic Chen systems are described, respectively, by the following equations:

$$\begin{cases} \dot{x} = a(y - x) + ayz, \\ \dot{y} = dx - cxz + y + w, \\ \dot{z} = xy - bz, \\ \dot{w} = -ry \end{cases} \tag{4}$$

where $a = 10,\ b = 3,\ c = 28,\ d = 17.717$ and $r = 12$, the chaotic attractor can been found. Description of hyperchaotic attractor are displayed in Fig. 1-3.

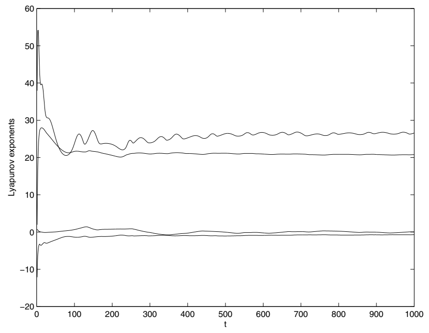


Fig. 1. Lyapunov exponents of system with $a = 10,\ b = 3,\ c = 28,\ d = 17.717$ and $r = 12$

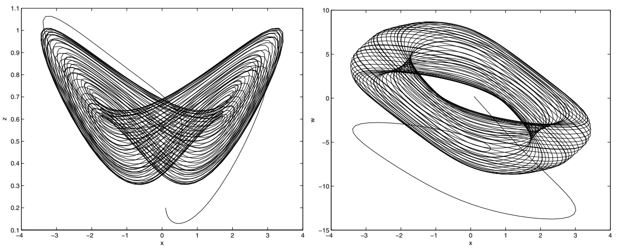


Fig. 2. The chaotic attractor of new Chen hyperchaotic system in $x - z$ plane and $x - w$ plane

We choose a master system as

$$\begin{cases} \dot{x}_m = a(y_m - x_m) + ay_m z_m, \\ \dot{y}_m = dx_m - cx_m z_m + y_m + w_m, \\ \dot{z}_m = x_m y_m - bz_m, \\ \dot{w}_m = -ry_m \end{cases} \tag{5}$$

and suppose that the slave system related to the master system with feedback controllers $u_i(i = 1, 2, 3, 4)$, which is given by

$$\begin{cases} \dot{x}_s = a(y_s - x_s) + a y_s z_s + u_1, \\ \dot{y}_s = d x_s - c x_s z_s + y_s + w_s + u_2, \\ \dot{z}_s = x_s y_s - b z_s + u_3, \\ \dot{w}_s = -r y_s + u_4 \end{cases} \tag{6}$$

where u_1, u_2, u_3 and u_4 are the control inputs. The aim of this section is to determine the controller $U = (u_1, u_2, u_3, u_4)^T$ for the chaos synchronization of two Chen hyperchaotic dynamical system.

Let the error state be $e(t) = (e_x(t), e_y(t), e_z(t), e_w(t))^T = (x_m + x_s, y_m + y_s, z_m + z_s, w_m + w_s,)^T$. Then, the error system can be written as

$$\begin{cases} \dot{e}_x = a(e_y - e_x) + a y_m z_m + a y_s z_s + u_1, \\ \dot{e}_y = d e_x - c x_m z_m - c x_s z_s + e_y + e_w + u_2, \\ \dot{e}_z = x_m y_m + x_s y_s - b e_z + u_3, \\ \dot{e}_w = -r e_y + u_4. \end{cases} \tag{7}$$

Theorem 1. *For the modified hyperchaotic Chen system, if the following feedback controllers $u_i(i = 1, 2, 3, 4)$ is chosen for the slave system, then the zero solution of the error system is globally stable, and thus globally anti-synchronization between the master system and the slave system was achieved.*

$$\begin{cases} u_1 = -a e_y - a y_m z_m - a y_s z_s, \\ u_2 = -d e_x + c x_m z_m + c x_s z_s - e_w - p e_y, \\ u_3 = -x_m y_m - x_s y_s, \\ u_4 = r(e_y - e_w) \end{cases} \tag{8}$$

where $p > 1$ is a positive constant.

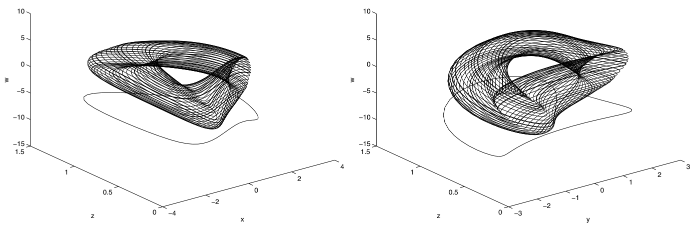


Fig. 3. The chaotic attractor of new Chen hyperchaotic system in $x-z-w$ subspace and $y-z-w$ subspace

Proof. Consider the following Lyapunov function

$$V = \frac{1}{2}(e_x^2 + e_y^2 + e_z^2 + e_w^2)$$

then the time derivative of V along the solution of error dynamical system gives that

$$\begin{aligned}\dot{V} &= \dot{e}_x e_x + \dot{e}_y e_y + \dot{e}_z e_z + \dot{e}_w e_w \\ &= e_x(a(e_y - e_x) + ay_m z_m + ay_s z_s + u_1) \\ &\quad + e_y(de_x - cx_m z_m - cx_s z_s + e_y + e_w + u_2) \\ &\quad + e_z(x_m y_m + x_s y_s - be_z + u_3) \\ &\quad + e_w(-re_y + u_4) \\ &= -(e_x,\ e_y,\ e_z,\ e_w)Q(e_x,\ e_y,\ e_z,\ e_w)^T\end{aligned} \quad . \tag{9}$$

Where $Q = \mathrm{diag}(a,\ p-1,\ b,\ r)$. Since V is positive definite and $\dot{V}$ is negative definite in the neighborhood of zero solution of error system (7), it follows that $lim_{t\to\infty}\|e(t)\| = 0$. Therefore, response system can globally anti-synchronize drive system asymptotically. This completes the proof.

4 Numerical Simulation Results

In this section, we present numerical results to verify the analytical predictions obtained in the previous section.

For the new hyperchaotic Chen system, if we choose $a = 10,\ b = 8/3,\ c = 28,\ d = 17$ and $r = 22$, anti-synchronization can been found in Fig. 4. and Fig. 5.

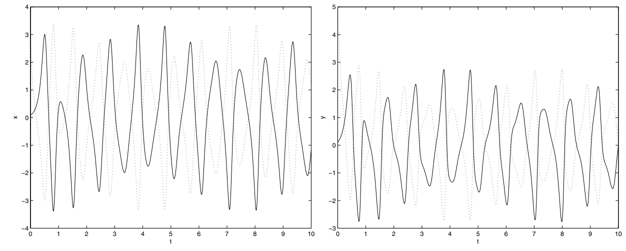


Fig. 4. Time response of states for drive system $(x_m,\ y_m,\ z_m,\ w_m)$ and response system $(x_s,\ y_s,\ z_s,\ w_s)$ with active control law activated. (a) Signals x_m and x_s; (b) signals y_m and y_s.

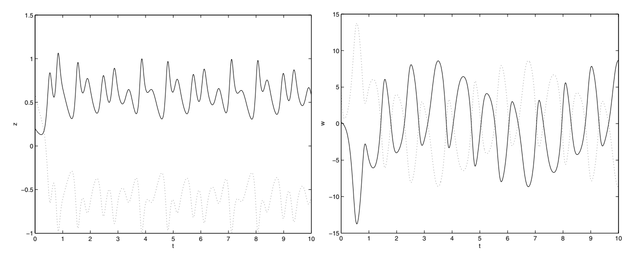


Fig. 5. Time response of states for drive system $(x_m,\ y_m,\ z_m,\ w_m)$ and response system $(x_s,\ y_s,\ z_s,\ w_s)$ with active control law activated. (c) signals z_m and z_s; (d) signals w_m and w_s.

5 Conclusion

In this paper, a new Chen hyperchaotic system was introduced and analyzed. Based on the Lyapunov stability theory, we propose a nonlinear control method to anti-synchronize two hyperchaotic systems. It has found that one can use control theory to synchronize and anti-synchronization hyperchaotic systems, and our proposed method has strong robustness. Numerical simulations are used to verify the effectiveness of the proposed control techniques.

Acknowledgment

The authors would like to thank these referees for their valuable suggestions and comments.

References

1. Pecora, L.M., Carroll, T.L.: Synchronization in chaotic systems. Phys. Rev. Lett. 64, 821–824 (1990)
2. Lu, W.L., Chen, T.P.: Synchronization of coupled connected neural networks with delays. IEEE Trans. Circuits and System-I 51, 2491–2503 (2004)
3. Lu, J., Cao, J.: Synchronization-based approach for parameters identification in delayed chaotic neural networks. Physica A 382, 672–682 (2007)
4. Cao, J., Wang, Z., Sun, Y.: Synchronization in an array of linearly stochastically coupled networks with time delays. Physica A 385, 718–728 (2007)
5. Sun, Y., Cao, J.: Adaptive lag synchronization of unknown chaotic delayed neural networks with noise perturbation. Physics Letters A 364, 277–285 (2007)
6. Cao, J., Lu, J.: Adaptive synchronization of neural networks with or without time-varying delays. Chaos 16, 013133 (2006)
7. Cao, J., Lu, J.: Adaptive complete synchronization of two identical or different chaotic (hyperchaotic) systems with fully unknown parameters. Chaos 15, 043901 (2005)
8. Amritkar, R.E.: Spatially synchronous extinction of species under external forcing. Phys. Rev. Lett. 96, 258102 (2006)
9. Al-Sawalha, M.M., Noorani, M.S.M.: Anti-synchronization of two hyperchaotic systems via nonlinear control. Commun. Nonlinear Sci. Numer. Simulat. 14, 3402–3411 (2009)
10. El-Dessoky, M.M.: Synchronization and anti-synchronization of a hyperchaotic Chen system. Chaos, Solitons and Fractals 39, 1790–1797 (2009)
11. Cheng, Z., Xin, Y., Li, X., Xing, J.: Synchronization and Lag Synchronization of Chaotic Networks. In: Yu, W., He, H., Zhang, N. (eds.) ISNN 2009. LNCS, vol. 5552, pp. 1197–1202. Springer, Heidelberg (2009)
12. Cheng, Z.: Globally exponential lag synchronization of complex networks. In: The Second International Conference on Information and Computing Science, vol. 3, pp. 376–378 (2009)
13. Cheng, Z.: New Chaos Produced from Synchronization of Chaotic Neural Networks. In: Sun, F., Zhang, J., Tan, Y., Cao, J., Yu, W. (eds.) ISNN 2008, Part I. LNCS, vol. 5263, pp. 40–46. Springer, Heidelberg (2008)
14. Lü, J., Yu, X., Chen, G.: Chaos synchronization of general complex dynamical networks. Physica A 334, 281–302 (2004)

15. Li, Y., Tang, S.K., Chen, G.: Generating hyperchaos via state feedback control. Int. J. Bifurcation and Chaos 15, 3367–3375 (2005)
16. Liao, X., Yu, P.: Analysis of the global exponent synchronization of Chuas circuit using absolute stability theory. Int. J. Birfurcation and Chaos 15, 3687–3881 (2005)
17. Nikolov, S., Clodong, S.: Occurrence of regular, chaotic and hyperchaotic behavior in a family of modified Rössler hyperchaotic systems. Chaos, Solitons and Fractals 22, 407–422 (2004)
18. Yan, Z., Yu, P.: Globally exponential hyperchaos (lag) synchronization in a family of modified hyperchaotic Rössler systems. Int. J. Bifurcat Chaos 17, 1759–1774 (2007)

Hopf Bifurcation Control for a Single Neuron Model with Delay-Dependent Parameters via State Feedback

Min Xiao

School of Mathematics and Information Technology, Nanjing Xiaozhuang University,
Nanjing 210017, China
candymanxm2003@yahoo.com.cn

Abstract. This paper deals with Hopf bifurcation control for a single neuron model with delay-dependent parameters. It has been shown that the system without control cannot guarantee a stationary state. As the range parameter of the system passes a critical value, Hopf bifurcation occurs early. To control the Hopf bifurcation, a state feedback controller is proposed to postpone the onset of undesirable Hopf bifurcation. Numerical simulation results confirm that the state feedback is efficient in controlling Hopf bifurcation.

Keywords: Hopf bifurcation control, Delay-dependent parameters, State feedback.

1 Introduction

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

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

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. 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. [2,3] considered (1) with parameter b depending on time delay τ described by

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

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. 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. A

© Springer-Verlag Berlin Heidelberg 2011

detailed analysis on the stability switches, Hopf bifurcation and chaos of (2) with delay-dependent parameters was given in [2,3]. Unlike in [2,3], where the delay τ was chosen as the bifurcation parameter, Xiao and Cao [4] used the range parameter a as bifurcation parameter. Critical values of Hopf bifurcation were assessed, and sufficient conditions for the bifurcated periodic solution were derived.

In recent years, we have witnessed rapidly growing interest in bifurcation control. Various methods have been used to control bifurcation [5]-[7]. The main contribution of this paper is design a general state feedback scheme to control Hopf bifurcation for (2). This state feedback controller keeps the equilibria of a system unchanged and also preserves the dimension of the system. Thus, the controlled system has the same structure as the original system. The results obtained are of general importance for optimizing the control technique of Hopf bifurcation and stimulate the search for further modifications aiming at the improvement of the control performance.

The organization of this paper is as follows. Sec. 2 is devoted to design a state feedback controller to control Hopf bifurcation for (2). In Sec. 3, numerical results are presented to verify the analytical predictions. Finally, conclusions are drawn in Sec. 4.

2 Hopf Bifurcation and Hopf Bifurcation Control

The results of Hopf bifurcation for the single neuron model with delay-dependent parameters, obtained in [4], are summarized here for comparison, completeness and convenience.

Theorem 1. *([4]) For (2), a Hopf bifurcation occurs from its trivial equilibrium, $x^* = 0$, when the range parameter, a, passes through the critical value, a^*.*

Here

$$a^* = \frac{\omega^*}{b(\tau)\sin(\omega^*\tau)}, \tag{3}$$

and

$$\omega^* \in (0, \frac{\pi}{\tau}), \tag{4}$$

is the root of

$$\omega\cot(\omega\tau) + \mu = \frac{\omega}{b(\tau)\sin(\omega\tau)}. \tag{5}$$

We now turn to design a state feedback controller in order to control the Hopf bifurcation arising from (2). Following the general idea of polynomial function controller [7], we propose a state feedback controller as follows for (2):

$$u(x) = -\alpha_1(x(t) - x^*) - \alpha_2(x(t) - x^*)^2 - \alpha_3(x(t) - x^*)^3. \tag{6}$$

α_1, α_2, and α_3 are feedback gain parameters, which can be manipulated to control the Hopf bifurcation to achieve desirable behaviors, such as delaying the onset of a Hopf bifurcation, changing the direction, stability and period of a Hopf bifurcation.

Remark 1. This nonlinear state feedback controller (6) preserves the equilibrium point x^* of (2). Thus, bifurcation control can be realized without destroying the properties of original system (2).

With the state feedback (6), the controlled model (2) becomes

$$\begin{aligned}\dot{x}(t) = & -\mu x(t) + a\tanh[x(t)] - ab(\tau)\tanh[x(t-\tau)] \\ & -\alpha_1(x(t)-x^*) - \alpha_2(x(t)-x^*)^2 - \alpha_3(x(t)-x^*)^3. \end{aligned} \tag{7}$$

Using Taylor expansion, we can expand the right-hand side of (7) around x^*, resulting in the following linearized equation:

$$\dot{x}(t) = (-\mu + a - \alpha_1)x(t) - ab(\tau)x(t-\tau), \tag{8}$$

which has the characteristic equation:

$$\lambda = -\mu + a - \alpha_1 - ab(\tau)e^{-\lambda\tau}. \tag{9}$$

Lemma 1. *If $\alpha_1 > 0$, then there exists a minimum positive number a_c^* such that*

(i) (9) has a pair of purely imaginary roots $\pm i\omega_c^*$ when $a = a_c^*$.
(ii) $a_c^* > a^*$, where a^* is defined by (3)-(5).

Here

$$a_c^* = \frac{\omega_c^*}{b(\tau)\sin(\omega_c^*\tau)}, \tag{10}$$

and

$$\omega_c^* \in (0, \frac{\pi}{\tau}), \tag{11}$$

is the root of

$$\omega\cot(\omega\tau) + \mu + \alpha_1 = \frac{\omega}{b(\tau)\sin(\omega\tau)}. \tag{12}$$

Proof. (i) If (9) has a pair of purely imaginary roots $\pm i\omega(\omega > 0)$, it is straightforward to obtain that

$$ab(\tau)\cos(\omega\tau) = a - \mu - \alpha_1, \qquad ab(\tau)\sin(\omega\tau) = \omega, \tag{13}$$

which yields (12). Solutions of (12) are the horizontal coordinates of the intersecting points between the curves $y = \omega\cot(\omega\tau) + \mu + \alpha_1$ and $y = \frac{\omega}{b(\tau)\sin(\omega\tau)}$. There are infinite number of intersecting points for these two curves that are graphically illustrated in Figure 1.

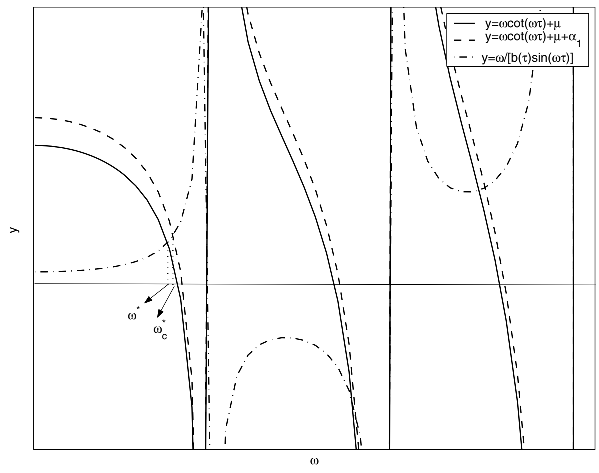


Fig. 1. Illustration for intersecting points between curves $y = \omega/[b(\tau)\sin(\omega\tau)]$ and $y = \omega\cot(\omega\tau) + \mu$ or $y = \omega\cot(\omega\tau) + \mu + \alpha_1$

Let ω_c^* satisfy (11) and be the root of (12), and define a_c^* as in (10). Then (a_c^*, ω_c^*) is a solution of (13). Thus $\pm i\omega_c^*$ is a pair of purely imaginary roots of (9) when $a = a_c^*$. It is easily seen form Figure 1 that ω_c^* is the minimum positive value among all horizontal coordinates of the intersecting points. So, a_c^* is the first value of $a > 0$ such that (9) has root appearing on the imaginary axis. The conclusion (i) follows.

(ii) It is clear from Figure 1 that ω^* is the horizontal coordinate of the intersecting point between the curves $y = \omega\cot(\omega\tau) + \mu$ and $y = \frac{\omega}{b(\tau)\sin(\omega\tau)}$, while ω_c^* is the horizontal coordinate of the intersecting point between the curves $y = \omega\cot(\omega\tau) + \mu + \alpha_1$ and $y = \frac{\omega}{b(\tau)\sin(\omega\tau)}$.

If $\alpha_1 > 0$ holds, we have $\omega\cot(\omega\tau) + \mu + \alpha_1 > \omega\cot(\omega\tau) + \mu$. Therefore, $\omega_c^* > \omega^*$. From the definitions of a^* and a_c^* in (3) and (10) respectively, we can obtain that a_c^* is larger than a^*. The conclusion (ii) follows. Then the proof is completed.

Theorem 2. *For the controlled system (7), there exists a Hopf bifurcation emerging from its equilibrium $x^* = 0$, when the range parameter, a, passes through the critical value, a_c^*, where the equilibrium point x^* is kept unchanged, and $a_c^* > a^*$ is defined by (10)-(12).*

3 Numerical Simulations

In this section, we present numerical results to verify the analytical predictions obtained in the previous section, using the state feedback controller to control the Hopf bifurcation of the single neuron model (2) with delay-dependent parameters. The numerical approach is based on a fourth-order Runge-Kutta integration scheme.

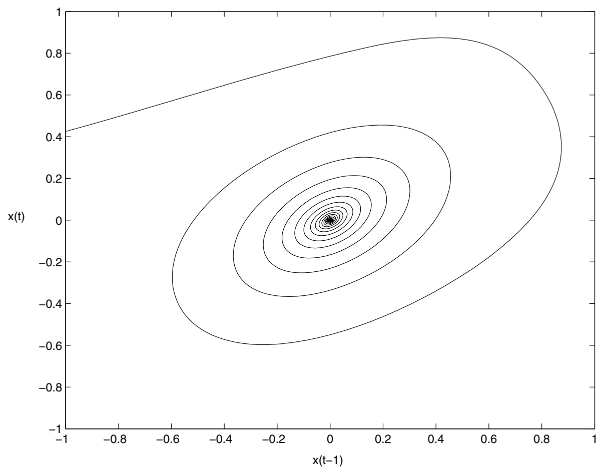


Fig. 2. Phase portrait of uncontrolled model (2) with $a = 0.68$

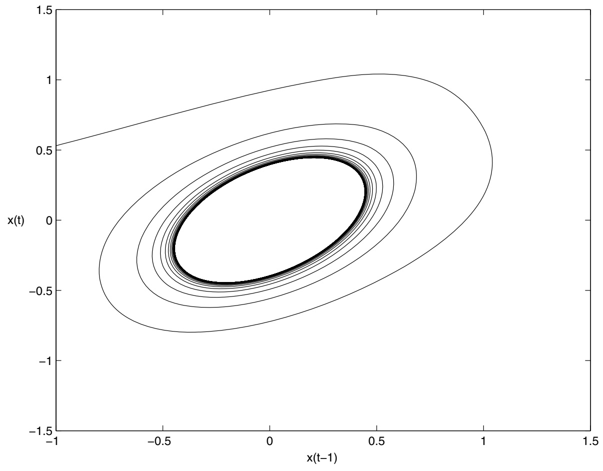


Fig. 3. Phase portrait of uncontrolled model (2) with $a = 0.77$

For a consistent comparison, the same model (2), used in [4], is discussed, with $b(\tau) = be^{-\alpha\tau}(b > 0, \alpha > 0)$, $\mu = 2, \tau = 2, b = 3$ and $\alpha = 0.12$. It follows from Theorem 1 that

$$a^* = 0.7351.$$

The dynamical behavior of this uncontrolled model (2) is illustrated in Figures 2-4. From Theorems 1, it is shown that when $a < a^*$, trajectories converge to the equilibrium point x^* (see Figure 2), while as a is increased to pass a^*, x^* loses its stability and a Hopf bifurcation occurs (see Figures 3 and 4).

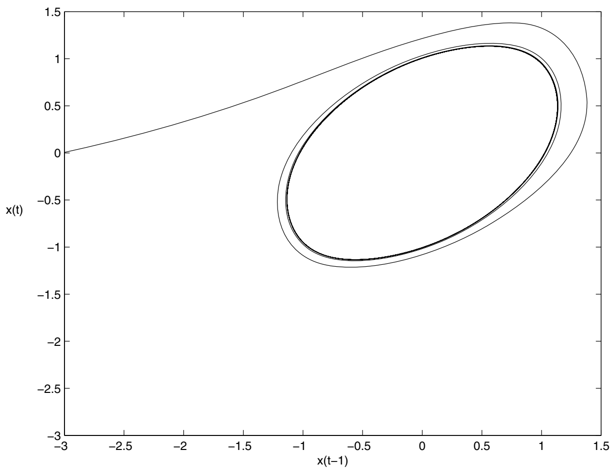


Fig. 4. Phase portrait of uncontrolled model (2) with $a = 0.95$

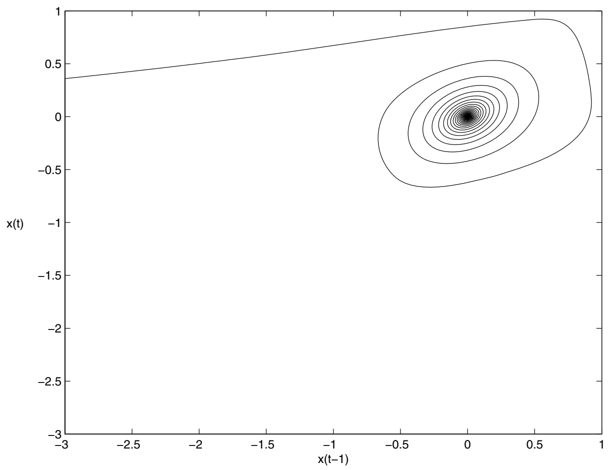


Fig. 5. Phase portrait of controlled model (7) with $a = 0.95$ and $\alpha_1 = 1, \alpha_2 = \alpha_3 = 0$

Now we choose appropriate values of α_1, α_2 and α_3 to control the Hopf bifurcation. It is easy to see from Theorem 2 that for linear state feedback control with a appropriate value of α_1, we can delay the onset of the Hopf bifurcation. For example, by choosing

$$\alpha_1 = 1, \quad \alpha_2 = 0, \quad \alpha_3 = 0,$$

we can apply Lemma 1 to obtain

$$a_c^* = 0.9618.$$

Note that the controlled model (7) has the same equilibrium point as that of the original model (2), but the critical value a^* increases from 0.7351 to 0.9618, implying that the onset of the Hopf bifurcation is delayed.

Under the linear state feedback control with $\alpha_1 = 1, \alpha_2 = \alpha_3 = 0$, we choose $a = 0.95 < a_c^*$, which is the same value as that used in Figure 4. According to Theorem 2, we conclude that instead of having a Hopf bifurcation, the controlled model (7) converges to the equilibrium point x^* as shown in Figure 5.

4 Concluding Remarks

We have discussed the effects of state feedback control upon the Hopf bifurcation for a single neuron model with delay-dependent parameters. By choosing appropriate control parameters α_1, it has been shown that the state feedback controller can effectively control Hopf bifurcation for the single neuron model with delay-dependent parameters. This state feedback controller is valid for any dynamical system close to the bifurcation point. We believe that theses results are of general importance for optimizing the control technique of Hopf bifurcation and will stimulate the search for further modifications aiming at the improvement of the control performance.

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. 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)
3. 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)
4. Xiao, M., Cao, J.D.: Range parameter induced bifurcation in a single neuron model with delay-dependent parameters. In: Zhang, L., Lu, B.-L., Kwok, J. (eds.) ISNN 2010. LNCS, vol. 6063, pp. 9–16. Springer, Heidelberg (2010)
5. Abed, E.H., Fu, J.H.: Local feedback stabilization and bifurcation control: I. Hopf bifurcation. Syst. Control Lett. 7, 11–17 (1986)
6. Chen, G.R., Moiola, J.L., Wang, H.O.: Bifurcation control: Theories, methods and applications. Int. J. Bifurc. Chaos 10, 511–548 (2000)
7. Yu, P., Chen, G.R.: Hopf bifurcation control using nonlinear feedback with polynomial functions. Int. J. Bifurc. Chaos 14, 1683–1704 (2004)

Changes in Electroencephalographic Power Spectra Associated with Reproductive Status in Frog

Guangzhan Fang, Jianguo Cui, Qin Chen, Ping Yang,
Jing Song, and Yezhong Tang

Chengdu Institute of Biology, Chinese Academy of Sciences
tangyz@cib.ac.cn

Abstract. Electroencephalogram (EEG) waveforms reflect the summed slow potentials generated by cortical neurons and can therefore be used to infer functional processes. In many species, individuals in the reproductive stage are more active and sensitive to species-typical stimuli than those in the non-reproductive stage. In the present study, changes in EEG power spectra were examined with respect to reproductive status in the music frog, *Babina daunchina*. The results indicated that in the reproductive stage, (1) power spectra of all EEG oscillations except for theta were significantly higher than in the non-reproductive stage; (2) for delta, significant increase of the power spectrum only appeared in the right hemisphere; and (3) the brain exhibited left-dominance of EEG spectra in the telencephalon and right-dominance of EEG spectra in the mesencephalon, providing evidence of lateralization of function. It is likely that sex hormone differences in the reproductive stage mediate these phenomena, through expression of their receptors, transcriptional regulatory elements, in specific regions of the forebrain and midbrain.

Keywords: EEG, power spectrum, reproductive stage, lateralization of function.

1 Introduction

In a physiological sense, electroencephalogram (EEG) power indicates the dynamics of electrical activity in populations of neurons and thus is assumed to reflect the capacity or performance of cortical information processing [1]. The power spectrum is strongly affected by a variety of specific factors such as age, arousal and the type of cognitive demands during task performance. For instance, within the alpha frequency range, EEG power is positively related to cognitive performance and brain maturity, whereas the opposite holds true for the theta frequency range [2]. Some hormones including the growth hormone, cortisol and its releasing factors, affect nocturnal sleep EEGs and have been studied thoroughly in both human and animal subjects [3],[4]. It is unclear to date how the hormones change the electrical characters and dynamics of neurons that finally result in alterations in EEG.

Reproductive status in vertebrates is mediated largely by environmental factors such as light-period and/or ambient temperature that initiate a hormone-produced cascade, i.e. activate the hypothalamic-pituitary-gonadal axis [5],[6]. The hypo-thalamus secretes and transports gonadotropin-releasing hormone (GnRH) to the pituitary which, in turn,

© Springer-Verlag Berlin Heidelberg 2011

secretes follicle-stimulating hormone (FSH) and luteinizing hormone (LH). The latter two hormones are delivered to whole body. Thereafter, sex hormones including estrogen, androgen and progesterone, are produced by gonadal cells after FSH and/or LH stimulation and delivered to all tissues including the brain. Sex hormones pass through the blood-brain barrier and bind to their receptors in the neuronal cytoplasm. The receptors move into the cellular nucleus after binding with their ligands, sex hormones, and act as transcriptional regulatory elements to reshape and/or rebuild neuronal structures, functions and behaviors partially or completely [7],[8],[9].

Animals including human beings behave differently in reproductive states when compared to non-reproductive states. They generally exhibit increased aggression, and are more sensitive to species-typical stimuli [10],[11]. This implies that the brain, at least some regions, experience anatomical and/or functional changes when animals become reproductive. We predicted that the changes in brain properties would be reflected by the EEG power spectra. In the present study, a species of frog, *Babina daunchina*, with complex acoustical behavior [12] was used to investigate changes in EEG power spectra in reproductive state induced by intraperitoneal injection of GnRH, and compared with those in non-reproductive state.

2 Materials and Methods

2.1 Animals

Ten adult music frogs (*Babina daunchina*), half males and half females, were captured from the Emei mountain area of Sichuan, China in July 2010. All subjects were housed by sex in two opaque plastic tanks (45×35 cm and 30 cm deep) containing mud and water. The tanks were placed in a room under controlled temperature conditions (23 ± 1°C) and maintained on a 12:12 light-dark cycle (lights on at 08:00 h). The animals were fed fresh live crickets every three days. Mean weights were 10.2 ± 2.4 g (mean ± SD), and frogs were 4.6 ± 0.3 cm in length at the time of surgery. All procedures used were approved by the Animal Care Committee of Chengdu Institute of Biology, Chinese Academy of Sciences.

2.2 Surgery

All experiments were conducted after the end of September 2010 when the reproductive season ended in music frogs [12]. The animals were deeply anesthetized by intraperitoneal injection of pentobarbital sodium (3 mg/100g) and the degree of anesthesia was evaluated through the toe pinch response.

Four bipolar cortical EEG electrodes, composed of miniature stainless steel screws (φ 0.8 mm), were implanted on the frog skull: the cerebellum (P), the left and right of telencephalon and mesencephalon (R1, R2, R3 and R4, the corresponding bipolar electrodes are abbreviated as PR1, PR2, PR3 and PR4, respectively) (Fig. 1). Twenty seconds of typical EEG waves were presented along with the corresponding bipolar electrodes (Fig. 1). R1 and R2 were implanted bilaterally 2.2 mm anterior to the lambda and 1.5 mm lateral to the midline, respectively, while R3 and R4 were implanted bilaterally 2.3 mm posterior to the lambda and 1.5 mm lateral to the midline. P was implanted 3.9 mm posterior to the lambda at the midline (Fig. 1).

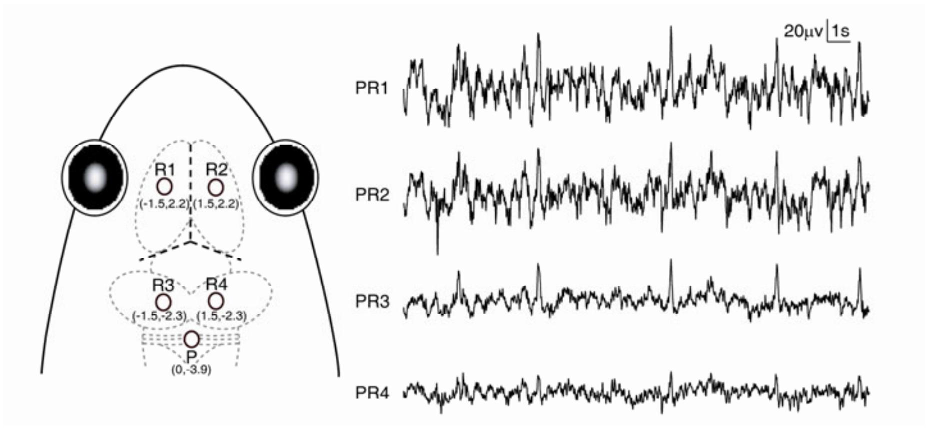


Fig. 1. Electrode placements and 20 s typical EEG tracings for different bipolar electrodes. The intersection of the three dashed lines in bold in the frog head denotes the position of the lambda.

All electrode leads, Formvar-insulated nichrome wires, were fixed on the skull of the frog with dental acrylic, and then welded to a light connector about 1 cm above the head of the animal. Each frog was housed singly for 2 days for recovery before the following experiments were performed. After the end of the experiments, all frogs were euthanized by overdose of intraperitoneal pentobarbital sodium and localizations of electrodes were confirmed.

2.3 Data Acquisition

The experiments were performed in a soundproof chamber in which the background noise was 23.0 ± 1.7 dB (mean ± SD) with an opaque plastic tank (80×60 cm and 55 cm deep) containing mud and water. Lights and temperature in the chamber were maintained as in the home-cages. A video camera with infrared light source was appended centrally above the tank for monitoring the behavior of the subject from outside of the chamber. Since the infrared light source was centrally positioned and the walls of the tank were symmetric, no optical cues were available for the subject's orientation. In order to eliminate effects of the state of alimentation on results, the subject was not fed during the experimental period.

The procedure for data collection consisted of electrophysiological and behavioral recordings in four sequential days: at the first day the subject was placed in the experiment tank and connected to the signal acquisition system (Chengyi, RM6280C; Sichuan, China) for habituation; neurophysiological data were collected on the subject in non-reproductive stage at the second day; the animal was administered with GnRH-6A (Sichuan, China, i.p. 1.25μg per frog at 21:30 h) at the third day [13],[14] and neurophysiological and behavioral data were acquired on the frog in reproductive stage with or without conspecific call playbacks at the fourth day. In the second and fourth day, the reproductive status of the subject was determined behaviorally by playback experiments, e.g. male calls and female phonotaxis in response to conspecific call playbacks.

Since subjects are nocturnal [12], especially in crepuscular hours, the signal acquisition system was set to record continuously from evening at 20:00 to next morning at 8:00. Bandpass filters, set to 0.16-100 Hz, were used for EEG signals with the notch filter of the amplifiers to eliminate possible interferences of 50 Hz. The sample frequency was set at 1000 Hz.

2.4 Data Processing

Before power spectrum analysis, for each frog and each bipolar electrode, representative waves (120 s) were selected randomly from the electrophysiological data acquired in the second day. The same time windows were adopted in selecting data for all four channels. These waves were divided into epochs of 5 s and standard deviation was calculated for each epoch and each channel.

Electrophysiological recordings of the first hour (20:00-21:00) during pre- and post- administration of GnRH-6A were used in the present study. After being band-pass filtered (0.5-45 Hz) and downsampled at 512 Hz, these waves were divided into epochs of 5 s and detrended for each epoch and each channel. Those epochs with their standard deviations were 3 times as large as the maximum of standard deviations of representative waves were discarded as artifacts. For artifact-free epochs, the power spectra of delta (0.5-5.5 Hz), theta (5.5-8.5 Hz), alpha (8.5-17 Hz) and beta (17-45 Hz) were computed by Welch's method with the Hamming window. The resolution of the power spectrum was set at 0.5 Hz and the boundaries of the EEG frequency bands were determined on the result of factor analysis of EEG in frogs (data not shown). Then the power spectra of each EEG band were averaged over the entire data length (1 h) for each channel, each frog and each day. Finally, these averaged data were further analyzed statistically. Because of many artifacts presented in PR3 and PR4 of one frog, data of nine frogs were included in statistical tests.

2.5 Statistical Analyses

Two factors, i.e. "electrode placement" and "reproductive status" were tested for each EEG band using 2-way within-subject ANOVA (i.e. 2-way repeated-measures ANOVA). Both main effects and interactions between factors were considered in the present study. Simple effects analysis would be applied when the interaction was significant, using the paired-samples *t*-test or the one-way repeated-measures ANOVA to the factor "electrode placement". For those showing significant differences by ANOVAs, the data were further analyzed for multiple comparisons using Least-significant difference (LSD) test. In both one-way and two-way ANOVAs, the values of epsilon (ε) of Greenhouse-Geisser were denoted when Greenhouse-Geisser correction was adopted. SPSS software (release 13.0) was utilized for the statistical analysis and a significance level of $p < 0.05$ was used in all comparisons.

3 Results

This paper presents exclusively electrophysiological data recorded from inactive or resting frogs before acoustic playbacks, and shows changes in the functional baseline of brain between non-reproductive and reproductive states.

For delta, the outputs of ANOVA revealed that main effects were statistically significant for the factors "electrode placement" (F (3,24) = 55.624; $p < 0.001$) and "reproductive status" (F (1,8) = 7.141, $\varepsilon = 1.000$; $p < 0.05$), respectively, and the interaction between factors was also statistically significant (F (3,24) = 5.132; $p < 0.05$). Because of the significant interaction between factors, simple effect analysis was further applied. When fixing the factor "electrode placement", the EEG power of delta band in reproductive stage was significant higher than those in non-reproductive stage for PR2 and PR4, the right hemisphere sites (paired samples t-test, $p < 0.05$; Fig. 2 A). No such significant difference of EEG power was found for PR1 and PR3, the left hemisphere sites (paired samples t-test, $p > 0.05$) between two reproductive stages. When fixing the factor "reproductive status", the EEG power of the telencephalon (PR1 and PR2) were significant higher than those of the mesencephalon (PR3 and PR4) either for the non-reproductive stage (F (3,24) = 54.116; $p < 0.001$; LSD, $p < 0.05$) or for the reproductive stage (F (3,24) = 34.642; $p < 0.001$; LSD, $p < 0.05$). Furthermore, in the reproductive stage, the EEG power of the left side of the telencephalon (PR1) was significantly higher than that of its right counterpart (PR2), while the EEG power of the right side of the mesencephalon (PR4) was significant higher than that of its left counterpart (PR3) (LSD, $p < 0.05$).

For theta, no significant difference of EEG power was found between the two reproductive stages (F (1,8) = 7.141, $\varepsilon = 1.000$; $p > 0.05$), and the main effect was statistically significant for the factor "electrode placement" (F (3,24) = 80.688; $p < 0.001$). Multiple comparisons indicated that the EEG power of the telencephalon (PR1 and PR2) was significant higher than that of the mesencephalon (PR3 and PR4) and, furthermore, the EEG power of the right side of the mesencephalon (PR4) was significant higher than that of the left counterpart (PR3) (LSD, $p < 0.05$; Fig. 2 B).

For alpha, the EEG power in the reproductive stage was significant higher than that in the non-reproductive state (F (1,8) = 14.024, $\varepsilon = 1.000$; $p < 0.05$); the main effect was significant for the factor "electrode placement" (F (3,24) = 58.575, $\varepsilon = 0.510$; $p < 0.001$). Multiple comparisons revealed that the EEG power of the telencephalon (PR1 and PR2) was significant higher than that of the mesencephalon (PR3 and PR4). The EEG power of the left side of the telencephalon (PR1) was significant higher than that of the right counterpart (PR2) while the EEG power of the right side of the mesencephalon (PR4) was significant higher than that of the left counterpart (PR3) (LSD, $p < 0.05$; Fig. 2 C).

For beta, the EEG power in the reproductive state was significant higher than that in the non-reproductive state (F (1,8) = 14.024, $\varepsilon = 1.000$; $p < 0.05$); the main effect was significant for the factor "electrode placement" (F (3,24) = 80.688, $\varepsilon = 0.520$; $p < 0.001$). Multiple comparisons showed that the EEG power of the telencephalon (PR1 and PR2) was significant higher than that of the mesencephalon (PR3 and PR4), while the EEG power of the left side of the telencephalon (PR1) was significantly higher than that of its right counterpart (PR2) (LSD, $p < 0.05$; Fig. 2 D).

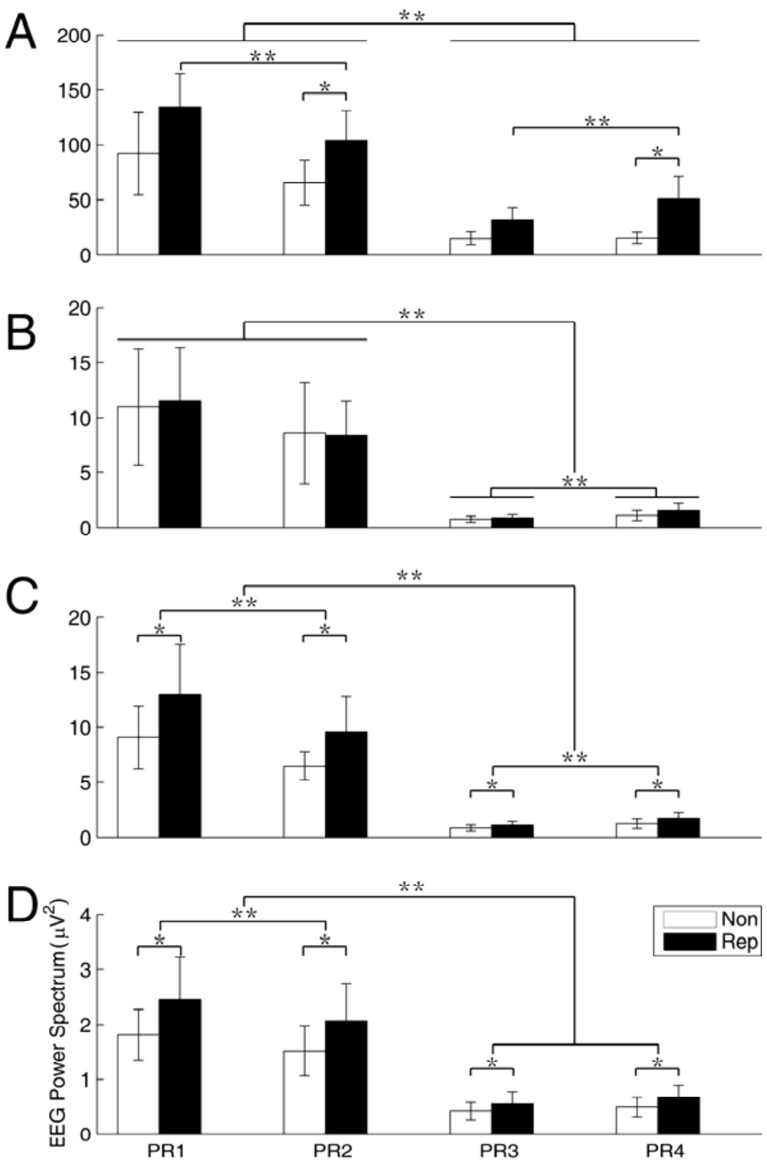


Fig. 2. The means and standard deviations of EEG power spectra for the two different reproductive stages, the four electrode locations and the four EEG bands (A, B, C and D for delta, theta, alpha and beta respectively). The symbols '*' and '**' denote that there are significant differences between the means of the EEG power on the both sides of '*' or '**'. * $p < 0.05$, ** $p < 0.001$. Abbreviations: Non, non-reproductive stage; Rep, reproductive stage; PR1, PR2, PR3 and PR4, the four bipolar electrodes.

4 Discussion

Vertebrates' behaviors differ dramatically between reproductive and non-reproductive states. While many mammals compete for mates by direct combat, almost all male frogs and toads as well as song birds contend through indirect ways, e.g. advertisement calls and visual displays. Female frogs and toads evaluate potential mates by listening to the advertisement calls [12],[15]. These behaviors, i.e. male calls and female responses to the calls, are often confined to the reproductive season, when sex hormone levels in the serum reach their peak [16],[17],[18]. Nevertheless, the sex hormone levels can be up- and down- regulated by gonadectomy and injection of upstream hormones, such as HCG or GnRH [13],[14]. Of our subjects, all male music frogs were mute before injecting GnRH and started to call about 12 hours after the injection. To the playback of conspecific calls, all females were unresponsive before injection and became responsive around 18 hours after the administration, frequently approaching the speaker broadcasting the calls (unpublished observations).

Behavioral changes along with the reproductive status imply neural modifications. Hormone receptors have been shown to mediate these changes [19],[20]. Thus, expressions of hormone receptors in the neurons of some nuclei and/or pathways relating to specific reproductive behaviors, should be an early event in the subsequent downstream cascade of neural modification [9],[21],[22]. Sex hormone receptors belong to the superfamily of transcriptional regulatory elements, and mediate hundred thousands of genes expressed [23],[24]. Changes in EEG power spectra might be one results of the neural changes associated with hormone receptor activation. Specifically for our data, increases in power of some EEG components might be related to males calling and females responding to the playbacks.

Anatomical and functional asymmetries in brains have been observed in many species including humans studied to date [25],[26]. Dramatic antero-posterior and left-right asymmetries of EEG power have been found for each EEG band in frogs (Fig. 2). These asymmetries are consistent with the previous observation of clear EEG differences between the two hemispheres and among different regions in both humans [27] and rodents [28],[29]. For all EEG bands of the frog brain, left-right asymmetries were mainly found in the left-dominance in the telencephalon and right-dominance in the mesencephalon (Fig. 2). These dominances may be bound up with the functions of the different brain regions. Hemispheric differences in overall modes of analysis of perceptual information, holding across different sensory modalities, have been suggested for humans, rats, pigeons and chickens [25]. Furthermore, mice, rats, primates, passerine birds and humans show a dominance of the left hemisphere in the production and/or perception of their species-specific vocalizations [25],[26]. Since acoustic communication is the most important form between individuals in frogs, the left-dominance of EEG power in the telencephalon might be resulted from the lateralization of function of the left hemisphere.

The right-dominance of EEG power in the mesencephalon of frogs is similar to other vertebrates [25], showing evolutionary conservation of brain functions. In addition, compared with those in non-reproductive stage, left-right asymmetries of EEG power were more extensive in reproductive stage for delta (Fig. 2 A). This extension implies that the brain may readily perceive and process the information correlated with reproduction. Considering the aforementioned left-right asymmetries, the possibility of lateralized expressions of the hormone receptors in different brain regions of frogs deserves attention.

Acknowledgments. The authors wish to thank Dr. Catherine E. Carr of Department of Biology and Dr. Steve E. Brauth of Department of Psychology, University of Maryland for their comments on the manuscript and English revising. This work was financially supported by grants from Chinese Academy of Sciences 'Bairenjihua' (KSCX2-YW-R-077) to Yezhong Tang, Chinese Academy of Sciences 'Xibuzhi-guang' (09C302) and the National Natural Science Foundation of China (30900141) to Jianguo Cui.

References

1. Tatum, W.O., Husain, A.M., Benbadis, S.R.: Handbook of EEG Interpretation. Demos Medical Publishing, New York (2008)
2. Klimesch, W.: EEG Alpha and Theta Oscillations Reflect Cognitive and Memory Performance: a Review and Analysis. Brain Research Reviews 29, 169–195 (1999)
3. Holsboer, F., von Bardeleben, U., Steiger, A.: Effects of Intravenous Corticotropin-Releasing Hormone upon Sleep-Related Growth Hormone Surge and Sleep EEG in Man. Neuroendocrinology 48, 32–38 (1988)
4. Steigera, A., Guldnera, J., Hemmeterb, U., Rothea, B., Wiedemanna, K., Holsboera, F.: Effects of Growth Hormone-Releasing Hormone and Somatostatin on Sleep EEG and Nocturnal Hormone Secretion in Male Controls. Neuroendocrinology 56, 566–573 (1992)
5. Francis, R.C., Soma, K., Fernald, R.D.: Social Regulation of the Brain-pituitary-gonadal Axis. Proc. Natl. Acad. Sci. USA 90, 7794–7798 (1993)
6. Vadakkadath, M.S., Atwood, C.S.: The Role of Hypothalamic-pituitary-gonadal Hormones in the Normal Structure and Functioning of the Brain. Cell. Mol. Life Sci. 62, 257–270 (2005)
7. Arnold, A.P., Breedlove, S.M.: Organizational and Activational Effects of Sex Steroids on Brain and Behavior: A reanalysis. Hormones and Behavior 19, 469–498 (1985)
8. Demotes-Mainard, J., Vernier, P., Vincen, J.D.: Hormonal Control of Neural Function in the Adult Brain. Current Opinion in Neurobiology 3, 989–996 (1993)
9. Tang, Y.Z., Piao, Y.S., Zhuang, L.Z., Wang, Z.W.: Expression of Androgen Receptor mRNA in the Brain of Gekko Gecko: Implications for Understanding the Role of Androgens in Controlling Auditory and Vocal Processes. Journal of Comparative Neurology 438, 136–147 (2001)
10. Trainor, B.C., Marler, C.: Testosterone, Paternal Behavior, and Aggression in the Monogamous California Mouse (Peromyscus californicus). Hormones and Behavior 40, 32–42 (2001)
11. Ball, G.F., Balthazart, J.: Hormonal Regulation of Brain Circuits Mediating Male Sexual Behavior in Birds. Physiology & Behavior 83, 329–346 (2004)
12. Cui, J.G., Wang, Y.S., Brauth, S.E., Tang, Y.Z.: A Novel Female Call Incites Male-female Interaction and Male-male Competition in the Emei Music Frog, *Babina Daunchina*. Animal Behaviour 80, 181–187 (2010)
13. Vignal, C., Kelley, D.B.: Significance of Temporal and Spectral Acoustic Cues for Sexual Recognition in *Xenopus Laevis*. Proceedings of the Royal Society B: Biological Sciences 274, 479–488 (2007)

14. Wang, Y.S., Cui, J.G., Yu, X.D., Tang, Y.Z.: Male Phonotropism and Answer Calls Elicited by Female Vocalizations in the African Clawed Frog, Xenopus laevis. Journal of Herpetology 44, 475–479 (2010)
15. Wells, K.D.: The Ecology and Behavior of Amphibians. University of Chicago Press, Chicago (2007)
16. Wetzel, D.M., Kelley, D.B.: Androgen and Gonadotropin Effects on Male Mate Calls in South African Clawed Frogs, *Xenopus laevis*. Hormones and Behavior 17, 388–404 (1983)
17. Tang, Y.Z., Zhuang, L.Z., Wang, Z.W.: Advertisement Calls and Their Relation to Reproductive Cycles in Gekko gecko (Reptilia, Lacertilia). Copeia, 248–253 (2001)
18. Medina, M.F., Ramos, I., Crespo, C.A., González-Calvar, S., Fernández, S.N.: Changes in Serum Sex Steroid Levels Throughout the Reproductive Cycle of *Bufo Arenarum* Females. General and Comparative Endocrinology 136, 143–151 (2004)
19. Lustig, R.H.: Sex Hormone Modulation of Neural Development in Vitro. Hormones and Behavior 28, 383–395 (1994)
20. Eisthen, H.L., Delay, R.J., Wirsig-Wiechmann, C.R., Dionne, V.E.: Neuromodulatory Effects of Gonadotropin Releasing Hormone on Olfactory Receptor Neurons. Journal of Neuroscience 20, 3947–3955 (2000)
21. Callard, G.V.: Androgen and Estrogen Actions in the Vertebrate Brain. Integrative and Comparative Biology 23, 607–620 (1983)
22. Kelley, D.B.: Auditory and Vocal Nuclei in the Frog Brain Concentrate Sex Hormones. Science 207, 553–555 (1980)
23. Evans, R.M.: The Steroid and Thyroid Hormone Receptor Superfamily. Science 240, 889–895 (1988)
24. Wahli, W., Martinez, E.: Superfamily of Steroid Nuclear Receptors: Positive and Negative Regulators of Gene Expression. The FASEB Journal 5, 2243–2249 (1991)
25. Bisazza, A., Rogers, L.J., Vallortigara, G.: The Origins of Cerebral Asymmetry: a Review of Evidence of Behavioural and Brain Lateralization in Fishes, Reptiles and Amphibians. Neurosci. Biobehav. Rev. 22, 411–426 (1998)
26. Walker, S.F.: Lateralization of Functions in the Vertebrate Brain: a Review. Br. J. Psychol. 71, 329–367 (1980)
27. Saastamoinen, A., Huupponen, E., Varri, A., Hasan, J., Himanen, S.L.: Systematic Performance Evaluation of a Continuous-scale Sleep Depth Measure. Med. Eng. Phys. 29, 1119–1131 (2007)
28. Vyazovskiy, V.V., Tobler, I.: Handedness Leads to Interhemispheric EEG Asymmetry During Sleep in the Rat. J. Neurophysiol. 99, 969–975 (2008)
29. Young, C.K., Mcnaughton, N.: Coupling of Theta Oscillations Between Anterior and Posterior Midline Cortex and with the Hippocampus in Freely Behaving Rats. Cereb. Cortex 19, 24–40 (2009)

Universal Analysis Method for Stability of Recurrent Neural Networks with Different Multiple Delays

Zhanshan Wang, Enlin Zhang, Kuo Yun, and Huaguang Zhang

School of Information Science and Engineering, Northeastern University, Shenyang, Liaoning, 110004, People's Republic of China
zhanshan_wang@163.com

Abstract. A universal stability analysis method on the basis of linear matrix inequality is proposed to solve the stability problem of recurrent neural networks with different kinds of multiple delays. Firstly, a universal neural networks model is analyzed to present a general framework for the stability study, in which a sufficient condition is derived. Secondly, by considering several special case of the universal model, a series of stability criteria are established, which have the same or similar structure and expression. All the obtained stability criteria present a general mode to study the stability of delayed dynamical systems.

Keywords: Recurrent neural networks, asymptotic stability, multiple delays, distributed delays, Lebesgue-Stiejies measures.

1 Introduction

Different kinds of recurrent neural networks with delays have been proposed and studied extensively in the literature [1-24]. Generally speaking, most of the studied delays in the neural networks can be resorted into two classes, concentrated/discrete delays and distributed. For the discrete or concentrated delays, they include constant delays and time varying delays. For example, constant delays τ, τ_j, τ_{ij} and their time varying counterparts. Distributed delays also include two classes, i.e., finitely distributed delays $\int_{t-\tau}^{t} g_j(x_j(s))ds$, $\int_{t-\tau_j}^{t} g_j(x_j(s))ds$, and their time varying counterparts, and infinitely distributed delays $\int_{-\infty}^{t} k_{ij}$ $(t-s)g_j(x_j(s))ds$ and $\int_{-\infty}^{t} k_j\ (t-s)g_j(x_j(s))ds$, where $g_j(\cdot)$ are the neuron activation functions, $k_{ij}(s)$ and $k_j(s)$ 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, algebraic inequality, spectral norm and linear matrix inequality (LMI) [1]-[10],[22].

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]-[20]. As pointed out in [10, 13], can we propose an effective approach to investigate them in a universal framework? An affirmative answer has been given in

© Springer-Verlag Berlin Heidelberg 2011

[10, 13] 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 [10, 13]. In our previous papers [23], we have established some LMI-based stability results for recurrent neural networks with different multiple delays. The main contribution is to decompose the connection matrix into several parts while keeping the nonlinear terms invariant. The advantage of that method is easy to combine several LMI-based stability results for the delayed neural networks in the existing literature. In contrast, we will decompose the nonlinear terms associated with its connection and keeping the original connection invariant in this paper, and establish some new stability criteria. The advantage of the proposed method is easy to unify many stability results into one framework, which will also give an affirmative answer to the problem proposed in [10, 13].

2 Problem Description and Preliminaries

The following recurrent neural networks with a general continuously distributed delays will be discussed,

$$\begin{aligned}\dot{u}_i(t) = &- a_i u_i(t) + \sum_{j=1}^{n} w_{ij} \int_0^{\infty} \bar{f}_j(u_j(t-s))dJ_{ij}(s) \\ &+ \sum_{j=1}^{n} w_{ij}^1 \int_0^{\infty} \bar{f}_j(u_j(t-\tau_{ij}-s))dK_{ij}(s) + U_i, \end{aligned} \tag{1}$$

where $u(t) = (u_1(t), \cdots, u_n(t))^T$, $A = \mathrm{diag}(a_1, \cdots, a_n)$, $a_i > 0$, $W = (w_{ij})_{n\times n}$ and $W_1 = (w_{ij}^1)_{n\times n}$ are real constant matrices, $\bar{f}(u(t)) = (\bar{f}_1(u_1(t)), \cdots, \bar{f}_n(u_n(t)))^T$, $\bar{f}_i(u_i(t))$ are the activation functions, τ_{ij} is the constant delay with $\tau_{ij} \le \tau_M$, U_i is the external constant input, $dJ_{ij}(s)$ and $dK_{ij}(s)$ are Lebesgue-Stieljies measures, $i, j = 1, \cdots, n$.

Assumption 1: The activation function $\bar{f}_i(x_i)$ is bounded and continuous, which satisfies $|\bar{f}_i(x_i)| \le F_i^b$, $0 \le (\bar{f}_i(\eta) - \bar{f}_i(v))/(\eta - v) \le \delta_i^f$, for any $\eta \ne v, \eta, v \in \Re$, and $\delta_i^f > 0$, $i = 1, \cdots, n$.

Assumption 2: The Lebesgue-Stieljies measures $dJ_{ij}(s)$ and $dK_{ij}(s)$ satisfy

$$\int_0^{\infty} dJ_{ij}(s) = \bar{m}_{ij} > 0, \int_0^{\infty} dK_{ij}(s) = m_{ij} > 0, \tag{2}$$

for some positive constants $\bar{m}_{ij} > 0$ and $m_{ij} > 0$, $i, j = 1, \cdots, n$. Let $\Delta = \mathrm{diag}(\delta_1^f, \cdots, \delta_n^f)$.

By the method shown in [18, 21, 20], there exists at least an equilibrium point for system (1). Let $u^* = [u_1^*, \ldots, u_n^*]^T$ be an equilibrium point of (1), and $x_i(t) = u_i(t) - u_i^*$, then model (1) is transformed into the following form,

$$\begin{aligned}\dot{x}_i(t) = -a_i u_i(t) &+ \sum_{j=1}^{n} w_{ij} \int_0^{\infty} f_j(x_j(t-s))dJ_{ij}(s) \\ &+ \sum_{j=1}^{n} w_{ij}^1 \int_0^{\infty} f_j(x_j(t-\tau_{ij}-s))dK_{ij}(s), \end{aligned} \tag{3}$$

where $f_j(x_j(t)) = \bar{f}_j(x_j(t) + u_j^*) - \bar{f}_j(u_j^*)$, $i, j = 1, \cdots, n$.

Inspired by the matrix-decomposition-method proposed in [1–3, 22], we decompose the general distributed delayed terms $\sum_{j=1}^{n} \int_0^{\infty} g_j(x_j(t-s))dJ_{ij}(s)$ and $\sum_{j=1}^{n} \int_0^{\infty} f_j(x_j(t-\tau_{ij}-s))dK_{ij}(s)$ in (3) and yields

$$\dot{x}(t) = -Ax + \sum_{k=1}^{n} \hat{D}_k \beta_k(t) + \sum_{k=1}^{n} \hat{E}_k \theta_k(t), \tag{4}$$

where $\hat{D}_k$ is an $n \times n$ matrix, whose k-th row is composed by the k-th row of matrix W, and the other rows are all zeros. $\hat{E}_k$ is an $n \times n$ matrix, whose k-th row is composed by the k-th row of matrix W_1, and the other rows are all zeros, $\beta_k(t) = \Big(\int_0^{\infty} f_1(x_1(t-s))dJ_{k1}(s), \cdots, \int_0^{\infty} f_n(x_n(t-s))dJ_{kn}(s) \Big)^T$, $\theta_k(t) = \Big(\int_0^{\infty} f_1(x_1(t-\tau_{k1}-s))dK_{k1}(s), \cdots, \int_0^{\infty} f_n(x_n(t-\tau_{kn}-s))dK_{kn}(s) \Big)^T$, $k = 1, \cdots, n$.

Remark 1. When $w_{ij} = w_{ij}^1 = 1$, system (1) was studied in [11, 15, 16, 10, 12, 14, 13, 17–19, 21, 20] via M-matrix method and algebraic inequality methods, respectively. However, no any LMI-based results for system (1) or its special cases have been published yet.

Remark 2. When the second term in system (3) is $\sum_{j=1}^{n} w_{ij} f_j(x_j(t))$, some LMI-based stability results have been proposed in [23], in which the interconnection coefficients are decomposed and keep the nonlinear terms invariant. Obviously, the model studied in this paper is mote complex and the decomposition method is different form that in [23], which can been seen in the sequel.

3 Main Results

Theorem 1. Suppose that Assumptions 1 and 2 hold. If there exist positive diagonal matrices P, D_i, H_i such that the following LMI holds,

$$\Psi = \begin{bmatrix} \Psi_{11} & \Psi_{\beta} & \Psi_{\theta} \\ * & \Psi_{22} & 0 \\ * & * & \Psi_{33} \end{bmatrix} < 0, \tag{5}$$

then the equilibrium point of system (4) is globally asymptotically stable, where $*$ denotes the symmetric part in a matrix, $\Psi_{11} = -2PA\Delta^{-1} + \sum_{i=1}^{n} \hat{D}_i \bar{M}_i + \sum_{i=1}^{n} H_i M_i$, $\Psi_{22} = -\mathrm{diag}(D_1\bar{M}_1^{-1}, \cdots, D_n\bar{M}_n^{-1})$, $\Psi_\beta = [P\hat{D}_1, \cdots, P\hat{D}_n]$, $\Psi_\theta = [P\hat{E}_1, P\hat{E}_2, \cdots, P\hat{E}_n]$, $\Psi_{33} = -\mathrm{diag}(H_1M_1^{-1}, H_2M_2^{-1}, \cdots, H_nM_n^{-1})$.

Proof. Let us consider the Lyapunov functional $V(t) = V_1(t) + V_2(t)$, where

$$V_1(t) = \sum_{i=1}^{n} 2p_i \int_0^{x_i(t)} f_i(s)ds + \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} \int_0^{\infty} dJ_{ij}(s) \int_{t-s}^{t} f_j^2(x_j(z))dz, \tag{6}$$

$$V_2(t) = \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij} \int_0^{\infty} dK_{ij}(s) \int_{t-\tau_{ij}-s}^{t} f_j^2(x_j(z))dz, \tag{7}$$

where $P = \mathrm{diag}(p_1, \cdots, p_n)$, $D_i = \mathrm{diag}(d_{i1}, \cdots, d_{in})$ and $H_i = \mathrm{diag}(h_{i1}, \cdots, h_{in})$ are positive diagonal matrix, $i = 1, \cdots, n$.

The derivatives of $V_1(t)$ and $V_2(t)$ are as follows,

$$\dot{V}_1(t) \leq 2f^T(x(t))P\dot{x}(t) + \sum_{i=1}^{n} f^T(x(t))D_i\bar{M}_i f(x(t)) - \sum_{i=1}^{n} \beta_i^T(t)D_i\bar{M}_i^{-1}\beta_i(t), \tag{8}$$

$$\dot{V}_2(t) \leq \sum_{i=1}^{n} f^T(x(t))H_iM_i f(x(t)) - \sum_{i=1}^{n} \theta_i^T(t)H_iM_i^{-1}\theta_i(t), \tag{9}$$

where $\bar{M}_i = \mathrm{diag}(\bar{m}_{i1}, \bar{m}_{i2}, \cdots, \bar{m}_{in})$, $M_i = \mathrm{diag}(m_{i1}, m_{i2}, \cdots, m_{in})$ are positive diagonal matrices, $i = 1, \cdots, n$.

By Assumption 1, the following condition holds,

$$-2f^T(x(t))PAx(t) \leq -2f^T(x(t))PA\Delta^{-1}f(x(t)). \tag{10}$$

Combining (8), (9) and (10), we have

$$\dot{V}(t) \leq \zeta^T\Psi\zeta < 0, \tag{11}$$

for any $\zeta = (f^T(x(t)), \beta_1(t), \cdots, \beta_n(t), \theta_1(t), \cdots, \theta_n(t))^T \neq 0$. Therefore, according to Lyapunov stability theory, the equilibrium point of system (4) is globally asymptotically stable if condition (5) holds.

Remark 3. From the proof of Theorem 1 we can see that the stability condition (2) is dependent on the information of Lebesgue-Stieljies measures $dJ_{ij}(s)$ and $dK_{ij}(s)$. However, this stability criterion is independent of the magnitude of time delays τ_{ij}, $i, j = 1, \cdots, n$.

4 Stability of Recurrent Neural Networks with Various Kinds of Multiple Delays

In this subsection, we will discuss four types of recurrent neural networks with various delays, which are all special cases of system (4).

1) In the case $dJ_{ij}(s) = \delta(s)\bar{w}_{ij}ds$, $dK_{ij}(s) = \delta(s)\bar{w}^1_{ij}ds$, where $\delta(s)$ is the Dirac-delta function, $\bar{w}_{ij}$ and $\bar{w}^1_{ij}$ are some constants, model (4) is reduced to the system with different multiple time varying delays,

$$\dot{x}_i(t) = -a_i x_i(t) + \sum_{j=1}^{n} w_{ij}\bar{w}_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} w^1_{ij}\bar{w}^1_{ij} f_j(x_j(t-\tau_{ij})), \tag{12}$$

which can be rewritten in the following form,

$$\dot{x}(t) = -Ax(t) + \sum_{i=1}^{n} \hat{D}_i \hat{F}_i(x(t)) + \sum_{i=1}^{n} \hat{E}_i \bar{F}(x(t-\bar{\tau}_i)), \tag{13}$$

where $\hat{F}_i(x) = (\bar{w}_{i1}f_1(x_1(t)), \cdots, \bar{w}_{in}f_n(x_n(t)))^T$, $\bar{F}(x(t-\bar{\tau}_i)) = (\bar{w}^1_{i1}f_1(x_1(t-\tau_{i1})), \cdots, \bar{w}^1_{in}f_n(x_n(t-\tau_{in})))^T$, $f(x(t)) = (f_1(x_1(t)), \cdots, f_n(x_n(t)))^T$, and the others are the same defined in (4).

Corollary 1. Suppose that Assumptions 1 and 2 hold. If there exist positive diagonal matrices P, D_i, H_i such that the following LMI holds,

$$\Psi_a^{'} = \begin{bmatrix} \Psi^a_{11} & \Psi_\beta & \Psi_\theta \\ * & \Psi^a_{22} & 0 \\ * & * & \Psi^a_{33} \end{bmatrix} < 0, \tag{14}$$

then the equilibrium point of system (13) is globally asymptotically stable, where $\Psi^a_{11} = -2PA\Delta_f^{-1} + \sum_{i=1}^n D_i\bar{W}_{wi} + \sum_{i=1}^n H_i\bar{W}^1_{wi}$, $\bar{W}^1_{wi} = \mathrm{diag}(\bar{w}^1_{i1}, \cdots, \bar{w}^1_{in})$, $\bar{W}_{wi} = \mathrm{diag}(\bar{w}_{i1}, \cdots, \bar{w}_{in})$, $\Psi^a_{44} = -\mathrm{diag}(D_1\bar{W}^{-1}_{w1}, D_2\bar{W}^{-1}_{w2}, \cdots, D_n\bar{W}^{-1}_{wn})$, $\Psi^a_{55} = -\mathrm{diag}\Big(H_1(\bar{W}^1_{w1})^{-1}, D_2(\bar{W}^1_{w2})^{-1}, \cdots, D_n(\bar{W}^1_{wn})^{-1}\Big)$, $\bar{W}^1_{wi} = \mathrm{diag}(\bar{w}^1_{i1}, \cdots, \bar{w}^1_{in})$, $\bar{W}_{wi} = \mathrm{diag}(\bar{w}_{i1}, \bar{w}_{i2}, \cdots, \bar{w}_{in})$, $i = 1, \cdots, n$, and the others are the same as those defined in Theorem 1.

Proof. Consider the following Lyapunov functional $V_s(t) = V_{s1}(t) + V_{s2}(t)$, where

$$V_{s1}(t) = \sum_{i=1}^{n} 2p_i \int_0^{x_i(t)} f_i(s)ds, \tag{15}$$

$$V_{s2}(t) = \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij} \int_{t-\tau_{ij}}^{t} \bar{w}^1_{ij} f_j^2(x_j(s))ds, \tag{16}$$

where $P_i = \mathrm{diag}(p_{i1}, \cdots, p_{in})$, $H_i = \mathrm{diag}(h_{i1}, \cdots, h_{in})$, $i = 1, \cdots, n$.

The derivatives of $V_{s1}(t)$ and $V_{s2}(t)$ are as follows, respectively,

$$\begin{aligned} \dot{V}_{s1}(t) \le & -2f^T(x(t))PA\Delta^{-1}f(x(t)) \\ & + 2f^T(x(t))P\Big[\sum_{i=1}^{n} \hat{D}_i\hat{F}_i(x(t)) + \sum_{i=1}^{n} \hat{E}_i\bar{F}(x(t-\bar{\tau}_i))\Big], \end{aligned} \tag{17}$$

$$\dot{V}_{s2}(t) = \sum_{i=1}^{n} f^T(x(t))H_i\bar{W}^1_{wi}f(x(t)) - \sum_{i=1}^{n} \bar{F}^T(x(t-\bar{\tau}_i))H_i(\bar{W}^1_{wi})^{-1}F(x(t-\bar{\tau}_i)). \tag{18}$$

Note that $\hat{F}_i(x(t)) = \bar{W}_{wi} f(x(t))$, then the following condition holds,

$$\sum_{i=1}^{n} \Big[f^T(x(t)) D_i \bar{W}_{wi} f(x(t)) - \hat{F}_i^T(x(t)) D_i \bar{W}_{wi}^{-1} \hat{F}_i(x(t)) \Big] = 0, \tag{19}$$

for any positive diagonal matrix $D_i, i = 1, \cdots, n$.

Combining (17), (18) and (19), we have

$$\dot{V}_s(t) \leq \vartheta_1^T(t) \Psi_a^{'} \vartheta_1(t) < 0, \tag{20}$$

for any $\vartheta_1(t) \neq 0$, $\vartheta_1(t) = \Big(f^T(x(t)), \hat{F}_1^T(x(t)), \cdots, \hat{F}_n^T(x(t)), \bar{F}^T(x(t-\bar{\tau}_1)), \cdots, \bar{F}^T(x(t-\bar{\tau}_n)) \Big)^T$. According to Lyapunov stability theory, the concerned system (13) is globally asymptotically stable.

2) In the case $dJ_{ij}(s) = \delta(s) \bar{w}_{ij} ds$, $dK_{ij}(s) = k_{ij}(s) \bar{w}_{ij}^1 ds$ and $\tau_{ij} = 0$, model (4) is reduced to the system with infinitely continuously distributed delays,

$$\dot{x}_i(t) = -a_i x_i(t) + \sum_{j=1}^{n} w_{ij} \bar{w}_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} w_{ij}^1 \bar{w}_{ij}^1 \int_0^{\infty} k_{ij}(s) f_j(x_j(t-s)) ds. \tag{21}$$

By the matrix decomposition method, system (21) can be rewritten as,

$$\dot{x}(t) = -Ax(t) + \sum_{j=1}^{n} \hat{D}_i \hat{F}_i(x(t)) + \sum_{j=1}^{n} \hat{E}_i \tilde{F}_i(t), \tag{22}$$

where $\tilde{F}_i(t) = \Big(\bar{w}_{i1}^1 \int_0^{\infty} k_{i1}(s) f_1(x_1(t-s)) ds, \cdots, \bar{w}_{in}^1 \int_0^{\infty} k_{in}(s) f_n(x_n(t-s)) ds \Big)^T$, $i = 1, \cdots, n$, and the others are the same as those defined in (13).

If the kernel function $k_{ij}(s)$ also satisfies

$$\int_0^{\infty} k_{ij}(s) ds = m_{ij} > 0, \tag{23}$$

then we have the following result for system (22).

Corollary 2. Suppose that Assumptions 1 and 2 hold. If there exist positive diagonal matrices P, D_i, H_i such that the following LMI holds,

$$\Psi_b^{'} = \begin{bmatrix} \Psi_{11}^b & \Psi_\beta & \Psi_\theta \\ * & \Psi_{22}^b & 0 \\ * & * & \Psi_{33}^b \end{bmatrix} < 0, \tag{24}$$

then system (22) is globally asymptotically stable, where $\Psi_{11}^b = -2PA\Delta_f^{-1} + \sum_{i=1}^{n} D_i \bar{W}_{wi} + \sum_{i=1}^{n} H_i \bar{W}_{wi}^1 M_i$, $\Psi_{22}^b = -\text{diag}(D_1 \bar{W}_{w1}^{-1}, \cdots, D_n \bar{W}_{wn}^{-1})$, $\Psi_{33}^b = -\text{diag}\Big(H_1 (\bar{W}_{w1}^1)^{-1} M_1^{-1}, \cdots, D_n (\bar{W}_{wn}^1)^{-1} M_n^{-1} \Big)$, and the others are the same as those defined in Corollary 1.

Proof. Consider the following Lyapunov functional $V_v(t) = V_{s1}(t)+V_{v2}(t)$, where $V_{s1}(t)$ is defined in (15), and

$$V_{v2}(t) = \sum_{i=1}^{n}\sum_{j=1}^{n} h_{ij}\bar{w}_{ij}^{1}\int_{0}^{\infty} k_{ij}(s)\int_{t-s}^{t} f_j^2(x_j(\theta))d\theta ds, \tag{25}$$

where $H_i = \mathrm{diag}(h_{i1}, \cdots, h_{in})$, $i = 1, \cdots, n$.

The derivatives of $V_{s1}(t)$ and $V_{v2}(t)$ are as follows,

$$\begin{aligned}\dot{V}_{s1}(t) \le & -2f^T(x(t))PA\Delta^{-1}f(x(t)) \\ & + 2f^T(x(t))P\Big[\sum_{j=1}^{n}\hat{D}_i\hat{F}_i(x(t)) + \sum_{j=1}^{n}\hat{E}_i\tilde{F}_i(t)\Big], \end{aligned}\tag{26}$$

$$\dot{V}_{v2}(t) \le \sum_{i=1}^{n} f^T(x(t))H_i\bar{W}_{wi}^{1}M_i f(x(t)) - \sum_{i=1}^{n}\tilde{F}_i^T(t)H_iM_i^{-1}(\bar{W}_{wi}^{1})^{-1}\tilde{F}_i(t). \tag{27}$$

Combining (19), (26) and (27), we have $\dot{V}_v(t) \le \vartheta_2^T\Psi_b^{'}\vartheta_2 < 0$ for $\vartheta_2 \ne 0$, where $\vartheta_2 = \Big(f^T(x(t)), \hat{F}_1^T(x(t)), \hat{F}_2^T(x(t)), \cdots, \hat{F}_n^T(x(t)), \tilde{F}_1^T(t), \tilde{F}_2^T(t), \cdots, \tilde{F}_n^T(t)\Big)^T$. According to Lyapunov stability theory, the system (22) is globally asymptotically stable.

3) In the case of 2), if the delay kernel function $k_{ij}(s)$ is 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 finitely distributed delays,

$$\dot{x}_i(t) = -a_ix_i(t) + \sum_{j=1}^{n} w_{ij}\bar{w}_{ij}f_j(x_j(t)) + \sum_{j=1}^{n} w_{ij}^{1}\bar{w}_{ij}^{1}\int_{t-r_{ij}}^{t} L_{ij}(t-s)f_j(x_j(s))ds. \tag{28}$$

If we further take a special form of the delay kernel function as $L_{ij}(s) = m_{ij}/r_{ij}$, then model (28) can be reduced to the following form,

$$\dot{x}_i(t) = -a_ix_i(t) + \sum_{j=1}^{n} w_{ij}\bar{w}_{ij}f_j(x_j(t)) + \sum_{j=1}^{n}\frac{w_{ij}^{1}\bar{w}_{ij}^{1}m_{ij}}{r_{ij}}\int_{t-r_{ij}}^{t} f_j(x_j(s))ds, \tag{29}$$

which can be rewritten in the following vector-matrix form,

$$\dot{x}(t) = -Ax(t) + \sum_{i=1}^{n}\hat{D}_i\hat{F}_i(x(t)) + \sum_{i=1}^{n}\hat{E}_i\check{F}_i(t), \tag{30}$$

where $\check{F}_i(t) = \big(\frac{\bar{w}_{i1}^{1}m_{i1}}{r_{i1}}\int_{t-r_{i1}}^{t} f_1(x_1(s))ds, \cdots, \frac{\bar{w}_{in}^{1}m_{in}}{r_{in}}\int_{t-r_{in}}^{t} f_n(x_n(s))ds, \big)^T$, and the others are the same defined in (13).

Corollary 3. If the condition in Corollary 2 holds, then the equilibrium point of system (30) is globally asymptotically stable.

Proof. Let us consider the Lyapunov functional $V_{vc}(t) = V_{s1}(t) + V_{vc2}(t)$, where $V_{s1}(t)$ is defined in (15), and

$$V_{vc2}(t) = \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{h_{ij}\bar{w}_{ij}^{1}}{r_{ij}}\int_{-r_{ij}}^{0}\int_{t+s}^{t} f_j^2(x_j(\theta))d\theta ds, \tag{31}$$

where $H_i = \text{diag}(h_{i1}, \cdots, h_{in})$, $i = 1, \cdots, n$.

The derivatives of $V_{s1}(t)$ and $V_{v2}(t)$ are as follows,

$$\begin{aligned}\dot{V}_{s1}(t) \leq & -2f^T(x(t))PA\Delta^{-1}f(x(t)) \\ & + 2f^T(x(t))P\Big[\sum_{i=1}^{n}\hat{D}_i\hat{F}_i(x(t)) + \sum_{i=1}^{n}\hat{E}_i\check{F}_i(t)\Big],\end{aligned} \tag{32}$$

$$\dot{V}_{vc2}(t) \leq \sum_{i=1}^{n} f^T(x(t))H_i\bar{W}_{wi}^{1}M_i f(x(t)) - \sum_{i=1}^{n}\check{F}_i^T(t)H_iM_i^{-1}(\bar{W}_{wi}^{1})^{-1}\check{F}_i(t), \tag{33}$$

where we have used the Jensen inequality $\int_{t-r_{ij}}^{t} f_j(x_j(s))ds \int_{t-r_{ij}}^{t} f_j(x_j(s))ds \leq r_{ij}\int_{t-r_{ij}}^{t} f_j^2(x_j(s))ds$, $i, j = 1, \cdots, n$.

Combining (19), (32) and (33), we have $\dot{V}_{vc}(t) \leq \vartheta_3^T\Psi_b^{'}\vartheta_3 < 0$ for $\vartheta_3 \neq 0$, where $\vartheta_3 = \Big(f^T(x(t)), \hat{F}_1^T(x(t)), \cdots, \hat{F}_n^T(x(t)), \check{F}_1^T(t), \check{F}_2^T(t), \cdots, \check{F}_n^T(t)\Big)^T$. According to Lyapunov theory, system (30) is globally asymptotically stable.

4) We generalize the model in the case of 3) to the following form,

$$\dot{x}_i(t) = -a_ix_i(t) + \sum_{j=1}^{n} w_{ij}\bar{w}_{ij}f_j(x_j(t)) + \sum_{j=1}^{n} w_{ij}^{1}\bar{w}_{ij}^{1}\int_{t-r_{ij}}^{t} f_j(x_j(s))ds, \tag{34}$$

where the parameters are the same as those defined in (4).

Corollary 4. Suppose that Assumptions 1 and 2 hold. If there exist positive diagonal matrices P, D_i, H_i such that the following LMI holds,

$$\Psi_d^{'} = \begin{bmatrix}\Psi_{11}^{d} & \Psi_{\beta} & \Psi_{\theta} \\ * & \Psi_{22}^{b} & 0 \\ * & * & \Psi_{33}^{d}\end{bmatrix} < 0, \tag{35}$$

then the equilibrium point of system (34) is globally asymptotically stable, where $\Psi_{11}^{d} = -2PA\Delta_f^{-1} + \sum_{i=1}^{n} D_i\bar{W}_{wi} + \sum_{i=1}^{n} H_i\bar{W}_{wi}^{1}R_i$, $\Psi_{22}^{b} = -\text{diag}(D_1\bar{W}_{w1}^{-1}, D_2\bar{W}_{w2}^{-1}, \cdots, D_n\bar{W}_{wn}^{-1})$, $\Psi_{33}^{d} = -\text{diag}\Big(H_1(\bar{W}_{w1}^{1})^{-1}R_1^{-1}, \cdots, D_n(\bar{W}_{wn}^{1})^{-1}R_n^{-1}\Big)$, and the others are the same as those defined in Corollary 1.

Proof. In a similar manner to the proof of Corollary 3, we can derive the results.

Remark 4. In short, the discrete delays $\tau_{ij}(t)$, distributed delays $\int_0^{\infty} k_{ij}(t-s)f_j(x_j(s))ds$ and $\int_{t-r_{ij}}^{t} f_j(x_j(s))ds$ can be included in the model (4) by choosing suitable kernel functions. For system (4) or its special cases, we can see that Corollaries 1- 4 have the similar structure of the stability criteria. Therefore, we provide a unified analysis method for system (4) and its special cases in the aspects of various delays.

5 Conclusions

Based on our previous studies on the recurrent neural networks with different multiple delays, we have proposed several LMI-based stability results for a class of recurrent neural network with different kinds of distributed delays. The proposed stability results are different from the existing ones and provide a universal analysis approach to the stability and stabilization of nonlinear systems with distributed delays.

Acknowledgments. This work was supported by the National Natural Science Foundation of China under Grants 61074073 and 61034005, by the Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant 200801451096, the Postdoctoral Science Foundation of China under Grant 200902547 and the Fundamental Research Funds for the Central Universities under grants N090404017 and N100104102.

References

1. Zheng, C.D., Zhang, H.G., Wang, Z.S.: An Augmented LKF Approach Involving Derivative Information of Both State and Delay. IEEE Trans. Neural Networks 21, 1100–1109 (2010)
2. Zhang, H.G., Wang, Z.S., Liu, D.R.: Global Asymptotic Stability of Recurrent Neural Networks with Multiple Time Varying Delays. IEEE Trans. Neural Networks 19, 855–873 (2008)
3. Zhang, H.G., Liu, Z.W., Huang, G.B., Wang, Z.S.: Novel Weighting-Delay-Based Stability Criteria for Recurrent Neural Networks with Time–Varying Delay. IEEE Trans. Neural Networks 21, 91–106 (2010)
4. Song, Q.K., Wang, Z.D.: Neural Networks with Discrete and Distributed Time Varying Delays: a General Stability Analysis. Chaos Solitons and Fractals 37, 1538–1547 (2008)
5. Yu, J., Zhang, K., Fei, S.M., 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)
6. Cao, J.D., Yuan, K., Li, H.X.: Global Asymptotic Stability of Recurrent Neural Networks with Multiple Discrete Delays and Distributed Delays. IEEE Trans. Neural Networks 17, 1646–1651 (2006)
7. Gopalsamy, K., He, X.Z.: Delay-Independent Stability in Bidirectional Asociative Neural Networks. IEEE Trans. Neural Networks 5, 998–1002 (1994)

8. Liang, J.L., Cao, J.D.: Global Output Convergence of Recurrent Neural Networks with Distributed Delays. Nonlinear Analysis: Real World Applications 8, 187–197 (2007)
9. Wang, Z.S., Zhang, H.G.: Global Asymptotic Stability of Reaction-Diffusion Cohen-Grossberg Neural Networks with Continuously Distributed Delays. IEEE Trans. Neural Networks 21, 39–49 (2010)
10. Chen, T.P.: Universal Approach to Study Delayed Dynamical Systems. In: Wang, L., Chen, K., S. Ong, Y. (eds.) ICNC 2005. LNCS, vol. 3610, pp. 245–253. Springer, Heidelberg (2005)
11. Chen, T.P., Lu, W.L.: Stability Analysis of Dynamical Neural Networks. In: IEEE Int. Conf. Neural Networks and Signal Processing, Nanjing, China, December 14-17 (2003)
12. Chen, T.P., Lu, W.L., Chen, G.R.: Dynamical Behaviors of a Large Class of General Delayed Neural Networks. Neural Computation 17, 949–968 (2005)
13. Chen, T.P.: Universal Approach to Study Delayed Dynamical Systems. Studies in Computational Intelligence 35, 85–110 (2007)
14. Chen, T.P., Lu, W.L.: Global Asymptotical 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)
15. Lu, W.L., Chen, T.P.: On Periodic Dynamical Systems. Chinese Annals of Mathematics Series B 25, 455–462 (2004)
16. Lu, W.L., Chen, T.P.: 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)
17. Lu, W.L., Chen, T.P.: Almost Periodic Dynamics of a Class of Delayed Neural Networks with Discontinuous Activations. Neural Computation 20, 1065–1090 (2008)
18. Liao, X.F., Li, C.D.: Global Attractivity of Cohen-Grossberg Model with Finite and Infinite delays. J. Mathematical Analysis and Applications 315, 244–262 (2006)
19. Liu, P.C., Yi, F.Q., Guo, Q., Yang, J., Wu, W.: Analysis on Global Exponential Robust Stability of Reaction–Diffusion Neural Networks with S-Type Distributed Delays. Physica D 237, 475–485 (2008)
20. Wang, L.S., Zhang, R.J., Wang, Y.F.: Global Exponential Stability of Reaction–Diffusion Cellular Neural Networks with S-Type Distributed Time Delays. Nonlinear Analysis: Real World Applications 10, 1101–1113 (2009)
21. Wang, L.S., Xu, D.Y.: Global Asymptotic Stability of Bidirectional Associative Memory Neural Networks with S-Type Distributed Delays. Internat. J. Syst. Sci. 33, 869–877 (2002)
22. Wang, Z.S., Zhang, H.G., Liu, D.R., 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)
23. Wang, Z.S., Zhang, H.G., Feng, J.: Stability Analysis of Recurrent Neural Networks with Distributed Delays Satisfying Lebesgue-Stieljies Measures. In: Zhang, L., Lu, B.-L., Kwok, J. (eds.) ISNN 2010. LNCS, vol. 6063, pp. 504–511. Springer, Heidelberg (2010)
24. Karimi, H.R., Gao, H.J.: New Delay-Dependent Exponential Synchronization for Uncertain Neural Networks with Mixed Time Delays. IEEE Trans. Systems, Man, and Cybernetics, B 40, 173–185 (2010)

A Class of New Generalized AOR Method for Augmented Systems[*]

Yu-xin Zhang, Heng-fei Ding[**], Wan-sheng He, and San-fu Wang

School of Mathematics and Statistics, Tianshui Normal University, Tianshui 741001, P.R. China
dinghf05@163.com

Abstract. In this paper, a class of new generalized AOR (GAOR) method with four parameters for augmented systems is established. This new method includes the SOR-Like method as a special case. The convergence of the new GAOR method for augmented systems is also studied. Numerical result is used to illustrate the efficiency of this new GAOR method.

Keywords: Saddle-point problem, AOR iteration method Augmented systems, Convergence.

1 Introduction

Consider the saddle point problem

$$\begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} x \\ c \end{pmatrix} = \begin{pmatrix} b \\ q \end{pmatrix}. \tag{1.1}$$

where $A \in R^{m\times m}$ is a symmetric positive definite matrix, and $B \in R^{m\times n}$ is a matrix of full column rank. This systems like (1.1) appears in many different applications of scientific computing. A large variety of methods for solving linear systems of the form (1.1) can be found in the literature [6,16,2,9,17], such as Uzawa-type schemes, SOR-Like method, GSOR method, HSS method, precondtioned Krylov subspace method and so on.

In this paper, we present a class of new generalized AOR (GAOR) method with four parameters for the augmented linear systems. The convergence of the GAOR method. Numerical result shows that the GAOR method may be more effective than the SOR-like method for solving the augmented linear system. And this GAOR method is further generalized for SOR-Like method.

This paper is organized as follows. In section 2, we establish the generalized AOR method for augmented system (1.1). In section 3, we study the convergence of the GAOR method. In section 4, some numerical examples are used to illustrate the efficiency of the new GAOR method.

[*] The project Supported by 'QingLan' Talent Engineering Funds and SRF(TSA0928) by Tianshui Normal University.

[**] Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

2 The Generalized AOR Method

For the coefficient matrix of the augmented system (1.1), we consider the following splitting

$$\mathbb{A} = \begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} = D - L - U \tag{2.1}$$

where

$$D = \begin{pmatrix} A & 0 \\ 0 & Q \end{pmatrix}, \quad L = \begin{pmatrix} 0 & 0 \\ -B^T & \alpha Q \end{pmatrix}, \quad U = \begin{pmatrix} 0 & -B \\ 0 & \beta Q \end{pmatrix}$$

and $Q \in R^{n\times n}$ is a nonsingular and symmetric matrix, $\alpha + \beta = 1$ and α, β are two nonzero real numbers. Let

$$\Omega = \begin{pmatrix} \omega_1 I_m & 0 \\ 0 & \omega_2 I_n \end{pmatrix}, \quad \Gamma = \begin{pmatrix} \gamma_1 I_m & 0 \\ 0 & \gamma_2 I_n \end{pmatrix},$$

where ω_1, ω_2 and γ_1, γ_2 are four nonzero positive real numbers, $I_m \in R^{m\times m}$ and $I_n \in R^{n\times n}$ are the m-by-m and the n-by-n identity matrices, respectively.

Denote $z^{(n)} = \begin{pmatrix} x^{(n)} & y^{(n)} \end{pmatrix}^T$ be the nth approximation of solution (1.1). Then our GAOR iteration scheme for solving the augmented linear system (1.1) is defined as follows:

$$z^{(n+1)} = \ell_\Omega z^{(n)} + (D - \Gamma L)^{-1}\Omega c, \tag{2.2}$$

where

$$\begin{aligned} z^{(n)} &= \ell_\Omega (D - \Gamma L)^{-1}[(I - \Omega)D + (\Omega - \Gamma)L + \Omega U] \\ &= \begin{pmatrix} (1-\omega_1)I_m & -\omega_1 A^{-1}B \\ \frac{\gamma_2\omega_1 - \omega_2}{1-\alpha\gamma_2} Q^{-1}B^T & I_n + \frac{\gamma_2\omega_1}{1-\alpha\gamma_2} Q^{-1}B^T A^{-1}B \end{pmatrix} \end{aligned} \tag{2.3}$$

is the iteration matrix of the GAOR method and

$$c = \begin{pmatrix} b \\ q \end{pmatrix}.$$

More precisely, the GAOR method is descripted as follows:

Given initial vector $x^{(0)} \in R^m$ and $y^{(0)} \in R^n$ and four relaxation parameters ω_1, ω_2 and γ_1, γ_2. For $k = 0, 1, 2, \ldots$ until the iteration sequence $\{((x^{(k)})^T, (y^{(k)})^T)^T\}$ is convergent, where

$$\begin{cases} y^{(k+1)} = y^{(k)} + \frac{\gamma_2\omega_1 - \omega_2}{1-\alpha\gamma_2} Q^{-1}B^T x^{(k)} + \frac{\gamma_2\omega_1}{1-\alpha\gamma_2} Q^{-1}B^T A^{-1}By^{(k)} + \frac{\omega_2}{1-\alpha\gamma_2} Q^{-1}q \\ -\frac{\gamma_2\omega_1}{1-\alpha\gamma_2} Q^{-1}B^T A^{-1}b, \\ x^{(k+1)} = (1-\omega_1)x^{(k)} - \omega_1 A^{-1}By^{(k)} + A^{-1}b, \end{cases}$$

and Q is an approximate (preconditioning) matrix of the Schur complement matrix $B^T A^{-1}B$.

Note that

$$(D - \Gamma L) = \begin{pmatrix} A & 0 \\ 0 & Q \end{pmatrix} - \Gamma \begin{pmatrix} 0 & 0 \\ -\gamma_2 B^T & \alpha\gamma_2 Q \end{pmatrix} = \begin{pmatrix} A & 0 \\ \gamma_2 B^T & 1 - \alpha\gamma_2 Q \end{pmatrix}. \tag{2.4}$$

and matrix A is a symmetric positive definite matrix and Q is a nonsingular matrix, we obtain $det(D-\Gamma L) = det(A)det(1 - \alpha\gamma_2 Q) = (1-\alpha\gamma_2)^n det(A)det(Q) \neq 0$, if and only if $1 - \alpha\gamma_2 \neq 0$, that is $\alpha\gamma_2 \neq 1$.

Obviously, when $\omega_1 = \omega_2 = \gamma_1 = \gamma_2$, the GAOR method reduces to the SOR-Like method; When $\omega_1 = \gamma_1$, $\omega_2 = \gamma_2$, the GAOR method becomes into the GSOR method; When $\omega_1 = \omega_2$, $\gamma_1 = \gamma_2$, the GAOR method reduces to the AOR method.

3 The Convergence of GAOR Method for Augmented Systems (1.1)

In this section, we study the convergence of the GAOR method for solving augmented system (1.1).

Throughout this paper, we always assume that $\omega_1, \omega_2 \neq 0$. Let λ be the eigenvalue of ℓ_Ω, and vector $\begin{pmatrix} x & y \end{pmatrix}^T$ be the corresponding eigenvalue of λ, then we have

$$\ell_\Omega z = \lambda z.$$

That is to say, it holds that

$$\begin{pmatrix} (1-\omega_1)I_m & -\omega_1 A^{-1}B \\ \frac{\gamma_2\omega_1 - \omega_2}{1-\alpha\gamma_2} Q^{-1}B^T & I_n + \frac{\gamma_2\omega_1}{1-\alpha\gamma_2} Q^{-1}B^T A^{-1} B \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \lambda \begin{pmatrix} x \\ y \end{pmatrix}. \tag{3.1}$$

which is equivalent to

$$\begin{cases} (1 - \omega_1 - \lambda)x = \omega_1 A^{-1}By, \\ \frac{\gamma_2\omega_1 - \omega_2}{1-\alpha\gamma_2} Q^{-1}B^T x + \frac{\gamma_2\omega_1}{1-\alpha\gamma_2} Q^{-1}B^T A^{-1}By + y = \lambda y. \end{cases} \tag{3.2}$$

Lemma 3.1. If λ is an eigenvalue of the matrix ℓ_Ω, then $\lambda \neq 1$.

Proof. If $\lambda = 1$ and $\begin{pmatrix} x & y \end{pmatrix}^T$ are the eigenvalue and the eigenvector of the matrix ℓ_Ω, respectively. It follows from equation (3.2) and $\omega_1 \neq 0$, that $-\omega_1 x = \omega_1 A^{-1}By$, $\frac{-\omega_2}{1-\alpha\gamma_2} Q^{-1}B^T x = 0$. We have $-Q^{-1}B^T A^{-1}By = 0$. Since the matrix $-Q^{-1}B^T A^{-1}B$ is nonsingular , we have $y = 0$. From equality $-\omega_1 x = \omega_1 A^{-1}By$, we have $x = 0$, which is a contradiction with the eigenvector of $\lambda = 1$, hence $\lambda \neq 1$.

Lemma 3.2. i) If $\gamma_2\omega_1 = \omega_2$, then $\lambda = 1 - \omega_1$ is least an m multiple eigenvalue of the matrix ℓ_Ω.
ii) If $\gamma_2\omega_1 \neq \omega_2$ and $m > n$, then $\lambda = 1 - \omega_1$ is least an $(m - n)$ multiple eigenvalue of the matrix ℓ_Ω.
iii) If $\gamma_2\omega_1 \neq \omega_2$ and $m = n$, then $\lambda = 1 - \omega_1$ is not the eigenvalue of the matrix ℓ_Ω.

Proof. From (2.3), if $\gamma_2\omega_1 = \omega_2$, then iterative matrix ℓ_Ω has least m eigenvalues $\lambda = 1-\omega_1$. This completes the proof of case one. For the last two cases, we always suppose that $\gamma_2\omega_1 \neq \omega_2$. If $\lambda = 1-\omega_1$ is the eigenvalue of the matrix ℓ_Ω, then there exists a nonzero vector $\begin{pmatrix} x & y \end{pmatrix}^T$, such that it satisfies (3.1). From (3.2) and $\omega_1 \neq 0$, we have $A^{-1}By = 0$ and $(\gamma_2\omega_1-\omega_2)Q^{-1}B^Tx = -\omega_1(1-\alpha\gamma_2)y$. Because the matrix B is column full rank, above equations are equivalent to $y = 0$ and $Q^{-1}B^Tx = 0$. With $Rank(B) = n$, if $m > n$, then $Q^{-1}B^Tx = 0$ has $m-n(>0)$ independent nonzero solutions. If $m = n$, then $Q^{-1}B^Tx = 0$ has no solution. This completes the proof of case 2 and case 3.

Theorem 3.3. i) If $m > n$, then $\lambda = 1-\omega_1$ is the eigenvalue of the matrix ℓ_Ω. ii) Denote by $J = -Q^{-1}B^TA^{-1}B$. If μ is an eigenvalue of the matrix J, then the λ determined by the equation (3.3)

$$(1-\omega_1-\lambda)(\lambda-1)(1-\alpha\gamma_2) = \omega_1(\lambda\gamma_2+\omega_2-\gamma_2)\mu, \tag{3.3}$$

is an eigenvalue of the matrix ℓ_Ω. On the other hand, if λ is an eigenvalue of the matrix ℓ_Ω such that $\lambda \neq 1$ and $\lambda \neq 1-\omega_1$, and μ satisfies (3.3), then μ is a nonzero eigenvalue of the matrix J.

Proof. From Lemma 3.2, we know that the first conclusion is obvious. Now we prove the second conclusion. Suppose that λ is an eigenvalue of the matrix ℓ_Ω , then the equation

$$\ell_\Omega z = \lambda z,$$

implies

$$\begin{pmatrix} (1-\omega_1)I_m & -\omega_1A^{-1}B \\ \frac{\gamma_2\omega_1-\omega_2}{1-\alpha\gamma_2}Q^{-1}B^T & I_n + \frac{\gamma_2\omega_1}{1-\alpha\gamma_2}Q^{-1}B^TA^{-1}B \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \lambda \begin{pmatrix} x \\ y \end{pmatrix}.$$

or equivalently,

$$\begin{cases} (1-\omega_1-\lambda)x = \omega_1A^{-1}By, \\ \frac{\gamma_2\omega_1-\omega_2}{1-\alpha\gamma_2}Q^{-1}B^Tx + \frac{\gamma_2\omega_1}{1-\alpha\gamma_2}Q^{-1}B^TA^{-1}By + y = \lambda y. \end{cases} \tag{3.4}$$

From the first equality in (3.4), we get

$$x = \frac{\omega_1}{1-\omega_1-\lambda}A^{-1}By,$$

and

$$[\frac{(\gamma_2\omega_1-\omega_2)\omega_1}{(1-\alpha\gamma_2)(1-\omega_1-\lambda)} + \frac{\gamma_2\omega_1}{1-\alpha\gamma_2}]Q^{-1}B^TA^{-1}By = (\lambda-1)y.$$

Suppose that μ is an eigenvalue of the matrix J, then we have

$$-[(\gamma_2\omega_1-\omega_2)\omega_1 + \gamma_2\omega_1(1-\omega_1-\lambda)]\mu = (1-\omega_1-\lambda)(\lambda-1)(1-\alpha\gamma_2).$$

Hence it follows that

$$(1-\omega_1-\lambda)(\lambda-1)(1-\alpha\gamma_2) = \omega_1(\lambda\gamma_2+\omega_2-\gamma_2)\mu.$$

We can prove the second assertion by reversing the process.

Corollary 3.1. If $\rho(\ell_\Omega)$ is the spectral radius of the matrix ℓ_Ω and $m > n$, then $\rho(\ell_\Omega) \geq |1 - \omega_1|$.

Remark: From the Corollary 3.1, it can be seen that if the GAOR method is convergent with $m > n$, then $0 < \omega_1 < 2$.

Lemma 3.3. (Young [11]) If b and c are real, then both roots of the quadratic equation $x^2 - bx + c = 0$ are less than one in modulus if and only if $|c| < 1$ and $|b| < 1 + c$.

Theorem 3.4. Let $A \in R^{m\times m}$ and $Q \in R^{n\times n}$ be symmetric positive definite and $B \in R^{m\times n}$ be of full column rank. Assume that all eigenvalues μ of the matrix J are real. Then if $\mu > 0$, the GAOR method is convergent when ω_1 satisfies $0 < \omega_1 < 2$ and ω_1, ω_2, γ_1, γ_2 satisfy the following conditions:

$$1 - \alpha\gamma_2 > 0, \quad \frac{\gamma_2 - \omega_2}{1 - \alpha\gamma_2} > -\frac{1}{\mu_{max}} \quad and \quad \frac{2\gamma_2 - \omega_2}{1 - \alpha\gamma_2} < \frac{4 - 2\omega_1}{\omega_1 \mu_{max}},$$

where μ_{max} is the largest eigenvalue of the matrix J.
Proof. From Theorem 3.1, we obtain that

$$\lambda^2 - (2 - \omega_1 - \frac{\omega_1\gamma_2}{1 - \alpha\gamma_2}\mu)\lambda + 1 - \omega_1 + \frac{(\omega_2 - \gamma_2)\omega_1}{1 - \alpha\gamma_2}\mu = 0.$$

By Lemma 3.1, we know that $|\lambda| < 1$ if and only if

$$|1 - \omega_1 + \frac{(\omega_2 - \gamma_2)\omega_1}{1 - \alpha\gamma_2}\mu| < 1, \tag{3.5}$$

and

$$|2 - \omega_1 - \frac{\omega_1\gamma_2}{1 - \alpha\gamma_2}\mu| < 2 - \omega_1 + \frac{(\omega_2 - \gamma_2)\omega_1}{1 - \alpha\gamma_2}\mu. \tag{3.6}$$

After some simple calculations, the above inequalities are transformed into the equivalent ones below:

$$\omega_1 - \frac{(\omega_2 - \gamma_2)\omega_1}{1 - \alpha\gamma_2}\mu > 0, \tag{3.7a}$$

$$2 - \omega_1 + \frac{(\omega_2 - \gamma_2)\omega_1}{1 - \alpha\gamma_2}\mu > 0, \tag{3.7b}$$

$$\frac{\omega_1\gamma_2}{1 - \alpha\gamma_2}\mu > \frac{(\gamma_2 - \omega_2)\omega_1}{1 - \alpha\gamma_2}\mu, \tag{3.7c}$$

$$2 - \omega_1 - \frac{\omega_1\gamma_2}{1 - \alpha\gamma_2}\mu + 2 - \omega_1 + \frac{(\omega_2 - \gamma_2)\omega_1}{1 - \alpha\gamma_2}\mu > 0. \tag{3.7d}$$

If $0 < \omega_1 < 2$, $1 - \alpha\gamma_2 > 0$, then (3.7c) holds stably. The relation (3.7a) is transformed into the following inequality:

$$\frac{\gamma_2 - \omega_2}{1 - \alpha\gamma_2} > -\frac{1}{\mu}, \tag{3.8}$$

and the relation (3.7b) and (3.7d) are transformed into the equivalent ones below:

$$\frac{2\gamma_2 - \omega_2}{1 - \alpha\gamma_2} < \frac{4 - 2\omega_1}{\omega_1\mu}, \tag{3.9}$$

and

$$-\frac{1}{\mu} < -\frac{1}{\mu_{max}}, \frac{4 - 2\omega_1}{\omega_1\mu} > \frac{4 - 2\omega_1}{\omega_1\mu_{max}}.$$

This completes the proof of Theorem 3.2.

Corollary. Suppose that μ is an eigenvalue of the matrix $-Q^{-1}B^TA^{-1}B$. If λ satisfies

$$(1 - \omega - \lambda)(\lambda - 1)(1 - \alpha\gamma) = \omega(\lambda\gamma + \omega - \gamma)\mu. \tag{3.10}$$

then λ is an eigenvalue of the matrix ℓ_Ω with $\omega_1 = \omega_2$, $\gamma_1 = \gamma_2$. Conversely, if λ is an eigenvalue of the matrix ℓ_ω such that $\lambda \neq 1$ and $\lambda \neq 1 - \omega$, and μ satisfies (3.10), then μ is a nonzero eigenvalue of the matrix $-Q^{-1}B^TA^{-1}B$.

4 Numerical Examples

In this section, we use a numerical example to further examine the effectiveness and show the advantages of the GAOR method over the SOR-like method. We compare the results of GAOR method with the results of the SOR-Like method [9]. These results show the effectiveness of the GAOR. We report the corresponding the number of iterations (denoted by IT) and the spectral radius (denoted by ρ) by choosing $Q = -B^TB$, with $\{((x^{(k)})^T, (y^{(k)})^T)^T\}$ being the finial approximate solution, end computation is $\frac{||r_k||_2}{||r_0||_2} < 10^{-6}$, where

$$r_k = \begin{pmatrix} b \\ q \end{pmatrix} - \begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} x^k \\ y^k \end{pmatrix}.$$

Example. [11] Consider the following example

$$\begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ q \end{pmatrix}. \tag{4.1}$$

where

$$A = (a_{ij})_{m\times m} = \begin{cases} i+1, & i = j, \\ 1, & |i-j| = 1, \\ 0, & \text{else.} \end{cases}$$

$$B = (b_{ij})_{m\times n} = \begin{cases} j, \ i = j + m - n, \\ 0, \qquad \text{else.} \end{cases}$$

We choose the right-hand-side vector $(b^T, q^T)^T \in R^{m+n} = (1.6, ..., 1.6, 1, 1, ..., 1)^T$.

In the following table, we list numerical results with respect to IT, ρ for the testing methods with respect to varying m and n.

Table 1. IT and spectral radius

$Methods$	m	n	α	β	ω_1	ω_2	γ_1	γ_2	IT	ρ
$SOR-Like$	50	40	0	1	1.8201	1.8201	1.8201	1.8201	292	0.9654
	200	150	0	1	1.9533	1.9533	1.9533	1.9533	1032	0.9903
	400	300	0	1	1.9759	1.9759	1.9759	1.9759	2066	0.9951
$GAOR$	50	40	0.32	0.68	1.6	5	1.67	1.6	46	0.7872
	200	150	0.5	0.5	1.6	8.5	1.67	1.6	41	0.7638
	400	300	0.6	0.4	1.6	4	1.67	1.6	34	0.7331
$GAOR$	50	40	0.32	0.68	1.6	1.9	1.67	1.6	134	0.9260
	200	150	0.5	0.5	1.6	1.9	1.67	1.6	208	0.9534
	400	300	0.6	0.4	1.6	1.9	1.67	1.6	81	0.8849

From above table, we can see that for augmented systems of linear equations, the iteration number in the GAOR method is less than that in the SOR-Like method. From the final two rows, we know that we can decrease the number of iterations by choosing suitable ω_2. The determination of optimum values of the parameters needs further studies, but we can infer that $\omega_2 > 2$.

References

1. Chun, C., Ham, Y.M.: Some sixth-order variants of Ostrowski root-finding methods. Appl. Math. Comput. 193, 389–394 (2007)
2. Arioli, M., Duff, I.S., de Rijk, P.P.M.: On the augmented system approach to sparse least squares problems. Numer. Math. 55, 667–684 (1989)
3. Bai, Z.Z., Parlett, B.N., Wang, Z.-Q.: On generalized successive overrelaxation methods for augmented linear systems. Numerische Mathematik (2005)
4. Darvishi, M.T., Hessari, P.: On convergence of the generalized AOR method for linear systems with diagonally dominant coefficient matrices. Appl. Math. Computk 176, 128–133 (2006)
5. Darvishi, M.T., Hessari, P.: Symmetrci SOR method for augmented systems. Appl. Math. Comput. (2006)
6. Darvishi, M.T., Khosro-Aghdam, R.: Symmetric successive overrelaxation methods for rank deficient linear systems. Appl. Math. Comput. 173, 404–420 (2006)
7. Elman, H., Golub, G.H.: Symmetric Inexact and precondioned Uzawa algorithms for saddle point problems. SIAM J. Numer. Anal. 31, 1645–1661 (1994)
8. Elman, H., Silvester, D.: Fast nonsymmetric iteration and preconditioning for Navier-Stokes equations. SIAM J. Sci. Comput. 17, 33–46 (1996)
9. Fischer, B., Ramage, A., Silvester, D.J., Wathen, A.J.: Minimum residual methods for augmented systems. BIT 38, 527–543 (1998)

10. Golub, G.H., Wu, X., Yuan, J.Y.: SOR-like methods for augmented systems. BIT 55, 71–85 (2001)
11. Hadjidimos, A., Yeyios, A.: The symmetric accelerated overrelaxation (SAOR) method. Math. Comput. Simulat. XXIV, 72–76 (1982)
12. Hu, Q.Y., Zou, J.: An iterative method with variable relaxation parameters for saddle-point problems. Math. SIMA J. Matrix Anal. Appl. 23, 317–338 (2001)
13. Santos, G.H., Silva, B.P.B., Yuan, J.Y.: Block SOR methods for rank deficient least squares problems. J. Comput. Appl. Math. 75 (1998)
14. Santos, G.H., Yuan, J.Y.: Preconditioned conjugate gradient methods for rank deficient least squares problems. Int. J. Comput. Math. 75 (1999)
15. Wright, S.: Stability of augmented system factorizations in interior point methods. SIAM J. Matrix Anal. Appl. 18, 191–222 (1997)
16. Young, D.M.: Iteratice solution for large linear systems. Academic Press, New York (1971)
17. Yuan, J.Y., Iusem, A.N.: Preconditioned conjugate gradient methods for generalized least squares problems. Comput. Appl. Math. 71, 287–297 (1996)
18. Zhang, G.F., Lu, Q.H.: On generalized SSOR method for augmented systems, Accepted manuscript. Accepted manuscript. J. Comput. Appl. Math. (2007)

Detecting the Topology of a Neural Network from Partially Obtained Data Using Piecewise Granger Causality

Xiaoqun Wu[1,2], Changsong Zhou[3], Jun Wang[1], and Jun-an Lu[1]

[1] School of Mathematics and Statistics, Wuhan University, 430072, Hubei
[2] State Key Laboratory of Software Engineering, Wuhan University, 430072, Hubei
[3] Department of Physics, Hong Kong Baptist University, Kowloon Tong, Hong Kong

Abstract. The dynamics and function of a network are influenced by the topology of the network. A great need exists for the development of effective methods of inferring network structure. In the past few years, topology identification of complex networks has received intensive interest and quite a few works have been published in literature. However, in most of the publications, each state of a multidimensional node in the network has to be observable, and usually the nodal dynamics is assumed known. In this paper, a new method of recovering the underlying directed connections of a network from the observation of only one state of each node is proposed. The validity of the proposed approach is illustrated with a coupled FitzHugh-Nagumo neurobiological network by only observing the membrane potential of each neuron and found to outperform the traditional Granger causality method. The network coupling strength and noise intensity which might also affect the effectiveness of our method are further analyzed.

Keywords: Complex networks, stochastic process, topology identification, Granger causality.

1 Introduction

Neural networks have attracted extensive attention in the past years. Often, these networks consist of a large number of neurons interacting with each other by synapses. The synaptic connections, representing a neural network's topological structure, is closely related to the intrinsic dynamics and functions of neurons. Therefore, understanding the interaction processes is of fundamental importance.

Complex networks of predefined topological structures have been broadly investigated, such as globally coupled networks, small-world networks, and scale-free networks. However, a network's topology is usually unknown or uncertain. In the past few years, topology identification of networks with coupled deterministic systems has received intensive research attention. In 2006, Yu et al. proposed a method for detecting the topology of complex networks based on chaotic synchronization[1]. In 2007, Tang et al. modified Yu's method and applied it to the topology identification of neural networks[2]. Later, Zhou and

© Springer-Verlag Berlin Heidelberg 2011

Lu[3] proposed a method to identify the topology of a general weighted complex network consisting of identical and different nodes. In 2008, Wu extended Zhou's work to a weighted network with coupling delay [4]; In 2010, Liu et al. put forward an approach for simultaneously identifying the topology and unknown parameters of uncertain general complex networks with time delay [5]. Shortly after, Zhou et al. presented a criterion to identify the topology of a coupled FitzHugh-Nagumo network by receiving the membrane potentials[6]. Recently, Zhao et al. proposed an adaptive topology identification rule using persistent exciting condition instead of previous linear independence condition on the orbit of synchronization manifold[7].

Nevertheless, most of the above-mentioned publications are based on a prerequisite, that is, all the states of each multi-dimensional node in a network have to be measurable[1]-[5], [7]. Particularly, the dynamical equation of each node in a network has to be known prior to successful identification[1]-[7]. However, regarding real-world applications, we can hardly know all the nodal dynamics and usually we can only observe or monitor one or two states of each node. Furthermore, system noise widely exists. Despite the tremendous efforts in revealing the topological effect on a variety of dynamics, how to infer the interaction pattern from dynamical data is still challenging as an inverse problem, especially in the absence of the knowledge of nodal dynamics and in the presence of noise.

From the viewpoint of signal processing, there are some techniques such as measuring the cross correlation or partial correlation among obtained time series to recover the interaction patterns among recorded signals. For example, Ren et al. presented a method based on measuring the dynamical correlation to predict the network topology [8]. However, these techniques only work for undirected network, which in many situations are not very satisfactory. To examine the directed connections in a network, one choice is to consider the causal influence one neural time series can exert on another. The basic idea can be traced back to Wiener[9] who conceived the notion that, if the prediction of one time series could be improved by incorporating the knowledge of a second one ,then the second series is said to have a causal influence on the first. Granger later formalized the prediction idea in the context of linear regression models[10]. The roles of the two time series can be reversed to address the question of causal influence in the opposite direction. Thus the interaction discovered in this way may be reciprocal or unidirectional. In the past few years, Granger causality has been widely employed in neuroscience and economics. Later on, some extensions have been proposed, such as the conditional Granger causality, partial Granger causality, blockwise Granger causality and so on[11]-[17].

Since Granger causality was originally formulated for linear models, its direct application to nonlinear systems may or may not be appropriate, depending on the specific problem. Some nonlinear methods based on Granger's idea have been proposed, such as the extended Granger causality theory by incorporating the embedding reconstruction technique for multivariate time series[14] and the kernel Granger causality[15]-[16]. However, the amount of data required for reliable analysis can be large and the algorithms and calculations are rather complicated.

In this paper, we aim to establish a simple and feasible approach to recover the interactions among simultaneously obtained nonlinear data from a neural network contaminated with noise, with only one sub-variable of each neuron is observed. Enlightened by the method of piecewise linear approximation, we partition obtained data into consecutive parts of identical length and apply Granger causality to each part. Next, we take the average of the lower bound of the 3σ confidence interval calculated from each part of data as the index for prediction. We compare the proposed method with traditional Granger causality using a synaptically coupled FigzHugh-Nagumo neural network model and demonstrate its validity and superiority. We further study the influence of the coupling strength and noise intensity on the proposed method and find it works well provided that the coupling strength is not so small and the noise is not so high as to obscure the effects.

2 Theory

2.1 Granger Causality

The method of detecting causal relations between two linear time series is based on linear prediction theory. For two stochastic processes X_t and Y_t that are jointly stationary, consider the following autoregressive prediction of the current value of $X(t)$ based on its past measurements:

$$X_t = \sum_{j=1}^{\infty} a_{1j} X_{t-j} + \epsilon_{1t}, var(\epsilon_{1t}) = \Sigma_1, \tag{1}$$

where ϵ_{1t} is the prediction error whose magnitude can be evaluated by its variance Σ_1. Then consider the following prediction of the current value of $X(t)$ based on its own past values as well as the past values of $Y(t)$:

$$X_t = \sum_{j=1}^{\infty} a_{2j} X_{t-j} + \sum_{j=1}^{\infty} b_{2j} Y_{t-j} + \epsilon_{2t}, var(\epsilon_{2t}) = \Sigma_2, \tag{2}$$

where Σ_2 represents the prediction accuracy of X_t using the previous values of both X_t and Y_t.

According to Wiener[9] and Granger[10], if the prediction of $X(t)$ improves by incorporating the past values of $Y(t)$, that is, $\Sigma_2 < \Sigma_1$ in some suitable sense, then we say that $Y(t)$ has a causal influence on $X(t)$. We quantify this causal influence by

$$F_{Y \to X} = ln \frac{\Sigma_1}{\Sigma_2}. \tag{3}$$

It is clear that $F_{Y \to X} = 0$ when there is no causal influence from Y to X and $F_{Y \to X} > 0$ when there is. Similarly, we may consider

$$Y_t = \sum_{j=1}^{\infty} c_{1j} Y_{t-j} + \eta_{1t}, var(\eta_{1t}) = \Gamma_1, \tag{4}$$

$$Y_t = \sum_{j=1}^{\infty} c_{2j} Y_{t-j} + \sum_{j=1}^{\infty} d_{2j} X_{t-j} + \eta_{2t}, var(\eta_{2t}) = \Gamma_2, \tag{5}$$

and say that $X(t)$ has a causal influence on $Y(t)$ if $\Gamma_2 < \Gamma_1$. Note that Eqs.(2) and (5) form the autoregressive model for vector time series, where similar techniques can be derived to detect the vectors' causal relations.

2.2 Conditional Granger Causality

The above analysis for two time series can be extended to more than two time series by analyzing them pairwise. However, pairwise analysis of more than two time series cannot detect indirect causal influences. For example, consider three time series X_t, Y_t and Z_t. Two possible causal interactions among them are shown in Fig. 1. In Fig. 1(a), the causal interaction from Y_t to X_t is indirect and mediated by Z_t. In Fig. 1(b), both direct and indirect causal interactions exist. Pairwise analysis would show an arrow from Y_t to X_t and thus cannot separate these two cases. Another possible causal connection is shown in Fig. 2, where

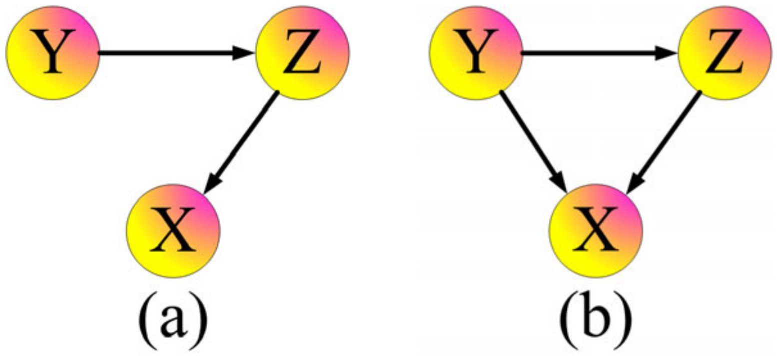


Fig. 1. Two patterns of causal relations: (a)Y drives X by way of Z; (b) There is a direct pathway from Y to Z

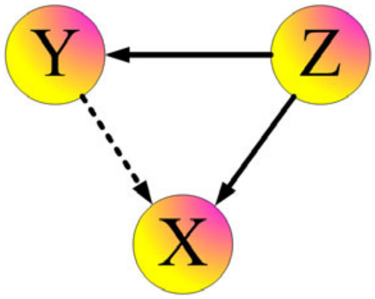


Fig. 2. The interaction from Y to X deduced by pairwise Granger Causality might be caused by the common drive Z

X_t and Y_t are simultaneously driven by Z_t. If the driving signal Z_t is powerful enough, X_t and Z_t might get into approximate synchronization and it is very likely that we get some causal connection between X_t and Y_t, such as the dash line from Y_t to X_t.

To examine whether the connection from Y_t to X_t is direct or mediated entirely by Z_t, we can use the following linear regression model:

$$\begin{cases} X_t = \sum_{j=1}^{\infty} a_{3j} X_{t-j} + \sum_{j=1}^{\infty} c_{3j} Z_{t-j} + \varepsilon_{1t}, var(\varepsilon_{1t}) = \Delta_1, \\ X_t = \sum_{j=1}^{\infty} a_{4j} X_{t-j} + \sum_{j=1}^{\infty} b_{4j} Y_{t-j} + \sum_{j=1}^{\infty} c_{4j} Z_{t-j} + \varepsilon_{2t}, var(\varepsilon_{2t}) = \Delta_2. \end{cases} \tag{6}$$

Define the conditional Granger causality index: $F_{Y \to X|Z} = ln\frac{\Delta_1}{\Delta_2}$. Thus, when the causal connection from Y_t to X_t is entirely mediated by Z_t, as in Fig. 1 or Fig. 2, $\{b_{4j}\}$ is uniformly zero and $\Delta_1 = \Delta_2$. Thus, we have $F_{Y \to X|Z} = 0$, which means that no further improvement in the prediction of X_t can be expected by including the past measurements of Y_t. On the other hand, when there is still a direct component from Y_t to X_t, the inclusion of past measurements of Y_t in addition to that of X_t and Z_t results in a better prediction of X_t, leading to $\Delta_2 < \Delta_1$ and $F_{Y \to X|Z} > 0$.

2.3 Piecewise Granger Causality

In real situations, especially for neural systems, all obtained time series should be nonlinear and a linear relationship as described above is only an approximation. Furthermore, usually only some sub-variable of a node is observable. In order to recover the underlying topology from obtained noise contaminated nonlinear time series, we propose the following approach to evaluate the directed connections:

1. For all the observed time series, partition them into K consecutive parts of identical length N_0.

2. Fit a linear regression model to the $i-$th partition of data $(i = 1, 2, ..., K)$ of two time series X_t and Y_t, as shown in (1) to (2) and calculate $F_{Y \to X}$. Using large sample method, we can calculate the $3\sigma-$ confidence interval, whose lower bound is denoted as $\alpha_i (i = 1, 2, ..., K)$. According to the 3σ principle for normal distributions, $F_{Y \to X}$ will fall within the $3\sigma-$ interval with probability 99.73%. Consequently, if $\alpha_i (i = 1, 2, ..., K)$ is greater than zero, we can conclude a directed connection $Y \to X$ from the $i-$th partition of data. Compute the Piecewise Granger causality index (PGCI) defined as $< \alpha_i >$, where $< \cdot >$ stands for averaging over the K partitions. If PGCI> 0, we have $Y \to X$.

3. Perform Step 2 for all the time series pairwise, hence we obtain an initial topological structure of the network. Nevertheless, there might be some false connections.

4. From the initial network structure, if there is a causal interaction as shown in Fig. 1(b) or Fig. 2, apply (6) to the K parts of data and similarly calculate the PGCI as defined in step (2), then we can eliminate those false causal connections which are indirect or mediated by a third time series.

Basically, N_0 cannot be too small, otherwise there is not enough information from each partition of data. Furthermore, large N_0 will make the partition useless. Thus for time series with fixed length, N_0 can be neither too small nor too large. K is thus determined according to the data length and N_0.

3 Numerical Simulations

In this section, a FigzHugh-Nagumo(FHN) network is used for illustration. The FitzHugh-Nagumo model

$$\begin{cases} \dot{V} = V - \frac{1}{3}V^3 - W + I_{ex}, \\ \dot{W} = 0.08(V + 0.7 - 0.8W) \end{cases} \tag{7}$$

is a two-dimensional simplification of the Hodgkin-Huxley model of spike generation in squid giant axons. Here, V is the membrane potential and W is the recovery variable, I_{ex} is the external stimulus current. The system was suggested by FitzHugh[18], who called it "Bonhoeffer-van der Pol model", and the equivalent circuit by Nagumo et al.[19].

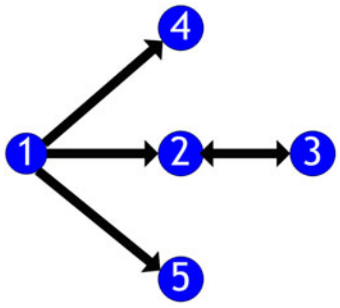


Fig. 3. The topology of the network

The network used in this paper is a dynamical network consisting of 5 synaptically coupled FHN neurons described as follows.

$$\begin{cases} \dot{V_i} = V_i - \frac{1}{3}V_i^3 - W_i + 0.1icos\frac{t}{50} + c\sum\limits_{j=1}^{5} a_{ij}(V_j - V_i) + \eta_i, \\ \dot{W_i} = 0.08(V_i + 0.7 - 0.8W_i), \qquad i = 1, 2, ..., 5. \end{cases} \tag{8}$$

Here η_i is the independent Gaussian white noise with zero mean and intensity σ that represents the noisy background. If there is a synaptic connection from

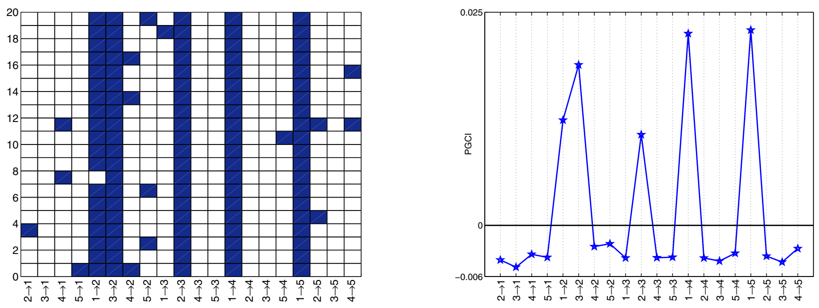


Fig. 4. Left: The results derived from the 20 parts of data for the 20 causal connections, where a blue rectangle on the $i-$th row representing $\alpha_i > 0$ for the possible connection on its corresponding $x-$label, and white otherwise. Right: Piecewise Granger causality index for the 20 causal connections. Here, the coupling strength $c = 8$, and the noise intensity $\sigma = 0.4$.

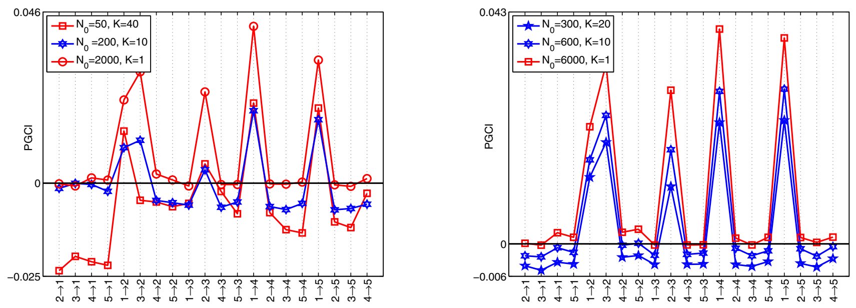


Fig. 5. Piecewise Granger causality index obtained from time series of different lengths, where the coupling strength is 8 and the noise intensity is 0.4. Left: The data length is 2000; Right: The data length is 6000.

the j-th neuron to the i-th neuron, $a_{ij} = 1$, otherwise $a_{ij} = 0$. c represents the synaptic connection weight. Generally, only the membrane potential V_i is observable. The underlying network topology used in our simulations is displayed in Fig. 3. Data used in the following simulations are generated from network(8) employing the Euler-Maruyama method with an equal time step 0.02. Unless otherwise stated, the time series generated are of length 6000 and partitioned into 20 consecutive parts of equivalent length, that is $N_0 = 300$ and $K = 20$.

The left figure in Fig. 4 displays the results derived from the K parts of data for the 20 possible connections labeled on the $x-$axis. The blue rectangles at the $i-$th$(i = 1, 2, ..., K)$ row represent that $\alpha_i > 0$ for the respective $x-$label connections, and white otherwise. The right figure illustrates the PGCI which is averaged over the K parts of data, as defined previously. It is clearly seen that the underlying topology in this network is correctly recovered.

To illustrate the validity of our proposed piecewise Granger causality, we apply the approach to data of different lengths and various partitions, as shown

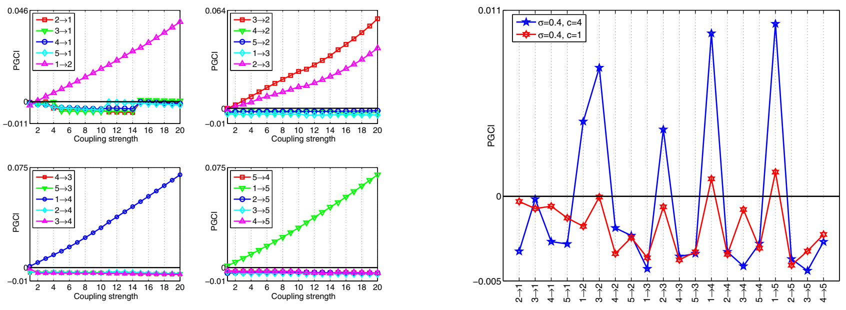


Fig. 6. Piecewise Granger causality index versus the coupling strength c, where the noise intensity σ is 0.4

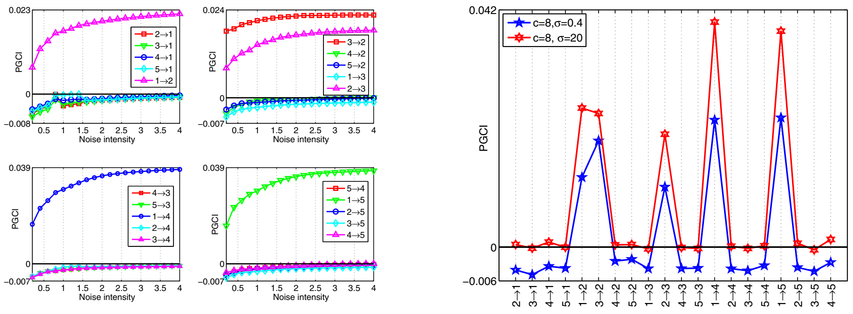


Fig. 7. Piecewise Granger causality index versus noise intensity σ, where the coupling strength is 8

in Fig. 5, where the blue curves represent correct topology detection and the red for false detection. The left figure presents the PGCI for time series of length 2000 with three different partitions. We can see that when $N_0 = 50, K = 40$, all the connections in the network are correctly recovered except the edge $3 \rightarrow 2$. The reason is that the data of each partition is so short that some connections may be missed. When N_0 is 200, our approach correctly recover the true topology. When there is no partition($N_0 = 2000, K = 1$), the approach goes back to a traditional one. However, some false causalities are given, such as the interactions $4 \rightarrow 1$, $4 \rightarrow 2$ and so on. The right figure shows the PGCI for time series of length 6000. When the data length of each partition is 300 or 600, our approach reveal the correct network structure, whereas the traditional Granger causality ($N_0 = 6000, K = 1$) tells many false positive connections. Hence we conclude that our proposed approach is valid for detecting network structure from partially obtained nonlinear time series provided that the data length for each partition is not too small, whereas the traditional Granger causality fail to do so.

To see the influence of network coupling strength c on the results, Fig. 6 shows the PGCI as a function of coupling strength c. From the left figure, we can obtain that as the coupling strength increases, our piecewise Granger causality method can correctly recovers the pattern of connectivity in our nonlinear model. When the coupling strength c is too small($c \leq 1$), some interactions may be missed, as shown in both figures. This is due to the reason that weak coupling makes the underlying topology obscured by noise and thus difficult to detect.

The result is also affected by the system noise intensity σ. Fig. 7 shows the PGCI varying with σ. From the tendency of the curves, we can predict that if σ gets too big, some false causal connections will appear. For example, when the noise intensity $\sigma = 20$, as shown by the red curve in the right figure, the causal interactions $4 \rightarrow 1$ and $4 \rightarrow 5$ are incorrectly given by our technique. This is because that too high noise have now obscured the underlying network topology.

4 Conclusions

In this paper, in order to recover the interactions among simultaneously obtained nonlinear data from a neural network contaminated with noise, a simple and feasible approach named Piecewise Granger Causality has been put forward. The validity and superiority of our technique has been demonstrated with a FigzHugh-Nagumo neural network with only the membrane potential of neurons is observable. The synaptic coupling strength and noise intensity have also been analyzed as two factors affecting the effectiveness of the piecewise Granger causality.

Acknowledgments

The authors would like to thank Professor Guanrong Chen for his insightful comments and kind suggestions. This work is supported in part by the Chinese National Natural Foundation(Grant Nos. 60804039 and 60974081), in part by the doctoral Fund of Ministry of Education of China (Grant No. 200804861072), and in part by State Key Laboratory of Software Engineering (Grant No. SKLSE2010-08-04).

References

1. Yu, D.C., Righero, M., Kocarev, L.: Estimating topology of networks. Phys. Rev. Lett. 97, 188701 (2006)
2. Tang, W.K.S., Yu, M., Kocarev, L.: Identification and monitoring of biological neural network. In: 2007 Proc. IEEE Int. Symp. Circuits Syst., pp. 2646–2649. IEEE Press, New York (2007)
3. Zhou, J., Lu, J.A.: Topology identification of weighted complex dynamical networks. Physica A 386, 481–491 (2007)
4. Wu, X.Q.: Synchronization-based topology identification of weighted general complex dynamical networks with time-varying coupling delay. Physica A 387, 997–1008 (2008)
5. Liu, H., Lu, J.A., Lü, J.H., Hill, D.J.: Structure Identification of Uncertain General Complex Dynamical Networks with Time Delay. Automatica-regular Papers 45, 1799–1807 (2009)

6. Zhou, J., Yu, W.W., Li, X.M., Small, M., Lu, J.A.: Identifying the topology of a coupled FitzHugh-Nagumo Neurobiological network via a pinning mechanism. IEEE Trans. Neural Networks 20, 1679–1684 (2009)
7. Zhao, J.C., Li, Q., Lu, J.A., Jiang, Z.P.: Topology identification of complex dynamical networks. Chaos 20, 23119 (2010)
8. Ren, J., Wang, W.X., Li, B.W., Lai, Y.C.: Noise bridges dynamical correlation and topology in complex oscillator networks. Phys. Rev. Lett. 104, 058701 (2010)
9. Wiener, N.: The theory of prediction. In: Beckenbach, E.F. (ed.) Modern Mathermatics for Engineers, ch. 8. McGraw-Hill, New York (1956)
10. Granger, C.W.J.: Investigating causal relations by econometric models and cross-spectral methods. Econometrica 37, 424–438 (1969)
11. Geweke, J.: Measurement of linear dependence and feedback between multiple time series. J. of the American Statistical Association 77, 304–313 (1982)
12. Geweke, J.: Measures of conditional linear dependence and feedback between time series. J. of the American Statistical Association 79, 907–915 (1984)
13. Ding, M., Bressler, S.L., Yang, W., Liang, H.: Short-window spectral analysis of cortical event-related potentials by adaptive multivariate autoregressive modelling: data preprocessing, model validation, and variability assessment. Biol. Cybern. 83, 35–45 (2000)
14. Chen, Y., Rangarajan, G., Feng, J., Ding, M.: Analyzing multiple nonlinear time series with extended Granger causality. Physics Letters A 324, 26–35 (2004)
15. Marinazzo, D., Pellicoro, M., Stramaglia, S.: Kernel method for nonlinear Granger Causality. Physical Review Letters 100, 144103 (2008)
16. Marinazzo, D., Pellicoro, M., Stramaglia, S.: Kernel method for nonlinear Granger Causality. Physical Review E 77, 056215 (2008)
17. Guo, S.X., Seth, A.K., Kendrick, K.M., Zhou, C., Feng, J.F.: Partial Granger causality-Eliminating exogeneous inputs and latent variables. J. of Neuroscience Methods 172, 79–93 (2008)
18. FitzHugh, R.: Impluses and physiological states in theoretical models of nerve membrane. Biophys. J. 1, 445–466 (1961)
19. Nagumo, J., Arimoto, S., Yoshizawa, S.: An active pulse transmission line simulating nerve axon. Proc. IRE 50, 2061–2070 (1962)

A Survey of Signal Propagation in Feedforward Neuronal Networks

Daqing Guo

School of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu 610054, People's Republic of China
dqguo07@gmail.com

Abstract. Understanding how neural activities are propagated through different brain regions is a critical and fundamental problem in neuroscience. A simple model for this type of signal propagation is the feedforward neuronal network, in which each neuron in a given layer only receives synaptic signals from neurons in its previous layer. This paper introduces and reviews the basic modeling framework about the signal propagation, two neural activities propagation modes, and several recent important results about signal propagation in the feedforward neuronal networks. Furthermore, a more generalized modeling framework based on unreliable synapses is also proposed and discussed.

Keywords: Signal propagation; synfire mode; firing rate mode; feedforward neuronal network; unreliable synapse.

1 Introduction

Human brain consists of tens of billion neurons and each neuron is connected to other neurons through more than 10000 synapses [1], [2]. It is known that the human brain might be the most complex matter in the universe. Such extreme complexity prohibits us to even think of a unified framework completely covering the human brain. However, over the past hundred years, biological research has accumulated an enormous amount of detailed knowledge about the structure and function of the brain. This provides us the possibility to build some specific models for fitting the existing neural data or explaining the observed experimental phenomena.

Signal propagation in the brain is a fundamental question that we face in the fields of experimental, theoretical, and computational neuroscience. Intensive biological experimental observations have revealed that many cognitive processing involves the propagation of neural activities through different brain regions [3]. One of the most used models to investigate this issue is the feedforward neuronal network, because it is not only easy to realize but also can be used to explain many neural activities propagation observed in experiments. In recent years, signal propagation in feedforward neuronal networks has been widely studied and several important neural activities propagation modes, such as the synfire mode [4]-[13] and firing rate mode [11]-[14], have been proposed. In this short review, we first introduce the basic modeling framework about the signal propagation. Then, we mainly review the synfire propagation and firing rate propagation, as well as several relevant results.

© Springer-Verlag Berlin Heidelberg 2011

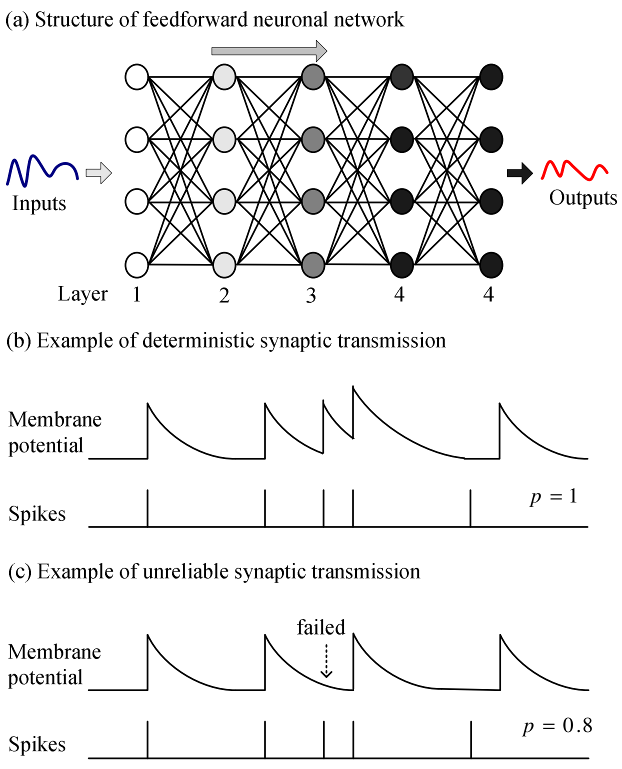

(a) Structure of feedforward neuronal network
Inputs
Outputs
Layer 1 2 3 4 4
(b) Example of deterministic synaptic transmission
Membrane potential
Spikes
$p = 1$
(c) Example of unreliable synaptic transmission
failed
Membrane potential
Spikes
$p = 0.8$

Fig. 1. Structure of the feedforward neuronal network (a), and examples of the deterministic synaptic transmission (b) and unreliable synaptic transmission (c)

Finally, we propose a more generalized modeling framework based on unreliable synapses and discuss several our recent numerical results.

2 Basic Modeling Framework

Information processing in the brain needs neural activities to be carried from one cortical area to the next. A simple model for this type of processing is a feedforward neuronal network [4], [12]. Consider the structure of a 5-layers feedforward neuronal network as shown in Fig. 1(a), in which each neuron in one layer feeds all synaptic connections to neurons in the next layer, and each neuron in the receiving layer is excited by all neuron in the previous layer. There is no feedback synaptic connection from neurons in downstream layers to neurons in upstream layers, and there is also no synaptic connection among neurons within the same layer. It should be emphasized that a typical feedforward neuronal network used in studying the signal propagation in the brain contains about 10 to 20 layers and each layer consists of at least 100 neurons. Due to computational complexity, the integrate-and-fire (IF) spiking neuron model is the most commonly used model to mimic the action potential firing dynamics of biological neurons. The subthreshold dynamics of the membrane potential of a single IF neuron can be expressed as follows [1], [2]:

$$\tau_m \frac{dV}{dt} = V_{\text{rest}} - V + RI(t) \ . \tag{1}$$

where V denotes the membrane potential and $I(t)$ represents the total input current. Each IF neuron is characterized by a membrane time constant τ_m , a resting membrane potential V_{rest}, and a membrane resistance R. The typical values for these parameters are: $\tau_m = 20$ ms, $V_{\text{rest}} = -60$ mV and $R = 20$ MΩ . In general, neurons can be divided into excitatory and inhibitory neurons. Excitatory neurons encourage the activity of neurons on which they act, while inhibitory neurons act in an opposite manner. Both excitatory and inhibitory neurons are considered to study the signal propagation in feedforward neuronal networks. Whenever the membrane potential of the IF neuron reaches a threshold V_{th} , the neuron fires a spike, and then the membrane potential is reset according to the resting potential, where it remains clamped for a 5 ms refractory period. The emitted spike is transmitted to the neurons in the adjacent downstream layer through synapses. The deterministic synaptic interaction model based on the pulse or α-function is considered in most relevant computational studies [5], [6], [8], [11], [12]. In this type of synaptic interaction scheme, the communication between neurons is assumed to be reliable, that is, a neuron transmits spikes to its postsynaptic neurons with successful transmission probability 1.

3 Signal Propagation in Feedforward Neuronal Networks

Two different modes of signal propagation have been proposed in recent years. The first is the synfire propagation mode, and the other is the firing rate propagation mode. In this section, we will review several important results about these two signal propagation modes.

3.1 Synfire Propagation

Synchronous of neuronal ensembles as well as rhythmic activities are widely found in various parts of the brain. An important view is that temporally precise neural information can be carried by a synchronous spike packet. If a group of synchronous spikes can be stably transmitted by subsequent groups of cortical neurons, it can provide a useful mode for propagating precise neural information [4]-[12]. This idea is the basis of neural activity propagation along a neuronal network with feedforward structure. Through numerical simulations, Diesmann et al. demonstrated that under certain conditions a synchronous firing state is indeed stable in a homogeneous feedforward neuronal network [5]-[9]. Once a number of synchronous spikes occur in one layer, neurons in their following layer might generate enough synchronous spikes again, and then such synchronous behavior is progressively propagated in deeper layers of the networks. This mode of neural activity transmission is called the synfire propagation. An elegant biological experiment based on the iteratively constructed

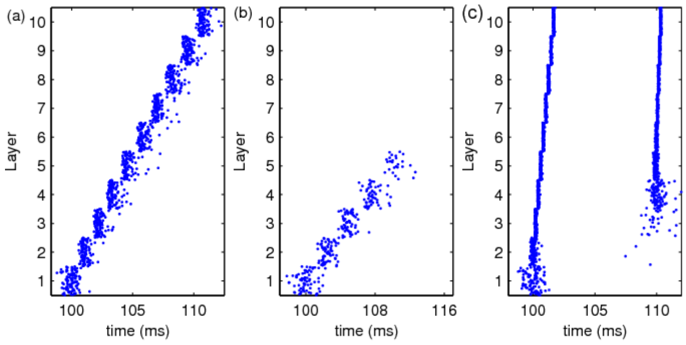


Fig. 2. Examples of synfire propagation. (a) Successful synfire propagation, (b) failed synfire propagation, and (c) synfire instability propagation. All subfigures are adapted from previously published papers [6], [9], [13].

feedforward network in vitro provides the direct evidence for the existence of the synfire propagation in cortical circuits [15].

There are several important factors determining whether the synchronous spike packet can be successfully and stably transmitted in the feedforward neuronal network. First, the survival rate of the synfire greatly depends on the intensity of the initial synfire packet [6], [8], [9]. For strong initial spike packet (large α and small σ, where α and σ are the number of spikes and temporal dispersion of the initial spike packet, respectively), the synfire activity is well built up after several initial layers and then this activity can be successfully transmitted along the entire network with high survival rate. By contrast, weak initial spike packet has the tendency to lead to the propagation of the neural activities becoming weaker and weaker with the increasing of layer number, and finally the neural activities are stopped before they reach the final layer of the network. To elucidate this phenomenon theoretically, the state space method was introduced to analysis the synchronous spiking in the feedforward neuronal network [6]-[9]. A stable attractor governing the synchronization dynamics was found to exist in the (α, σ) space. Within the basin of attraction, the survival probability of the synfire propagation is close to 1; otherwise, it drops to 0 rapidly. Noise also plays a key role in the synfire propagation [3], [8]. Although the temporal spread increases with noise, under certain conditions at an intermediate level of noise the basin of attraction achieves a maximum extent, which in fact is a phenomenon of stochastic resonance [8]. On the other hand, during the process of synfire propagation, a pronounced refractory behavior is required in order to prohibit each neuron firing more than once within one spike packet. If not so, the number of spikes per packet and the width might grow as the layer number grows. However, it should be noted that, with sufficiently strong synapses, a synchronous spike packet in one layer can still evoke a train of two or more synchronous spike packets in the next layer, even though the refractory period of neuron is considered. This will lead to the eruption of synfire propagation (we term this “synfire instability propagation” and see Refs. [3] and [13]) that should be avoided in the brain.

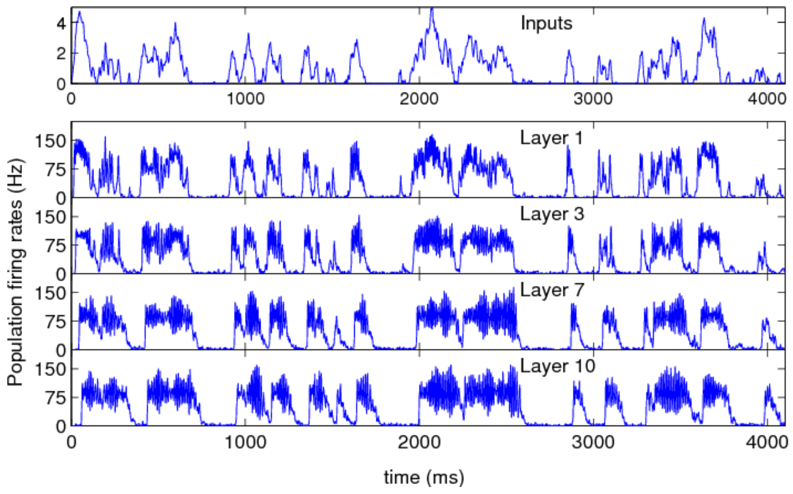


Fig. 3. A typical example of the firing rate propagation. The figure is adapted from previously published papers [14] and [16].

3.2 Firing Rate Propagation

One conventional view of neural coding is that the features of the input signal can be encoded in the population firing rate of neurons. Some researches have postulated that the feedforward neuronal network might also be a simple model of transmitting the asynchronous firing rates. Such mode of signal transmission is the so-called firing rate propagation [11]-[14]. It has been found that the input-rate to output-rate of the feedforward neuronal network may be linearized if neurons in the first layer are driven by a same external input signal and each neuron in the network is injected a suitable independent background noise (an example of firing rate propagation, see Ref. [17]). Noise is necessary for the firing rate propagation, which is used to prevent the occurrence of synchronization even within the groups of neurons carrying the signal [14]-[16]. Too weak noise will result in neurons in the same layer fire synchronous; while too strong noise will lead to the background firing overwhelm the external input signal. At an appropriate noise level, all neurons desynchronize and their membrane potentials are close to threshold. In this case, both the strong and weak components of the external input signal can be transmitted and the waveform of the signal is well encoded by the population firing rate, thus resulting in better propagation performance. Furthermore, the firing rate propagation requires not only appropriate background current but also well tuned synapses. If the network only consists of excitatory neurons, the excitatory synaptic strength will determine the gain of the propagation. When the synaptic strength is too weak, the response in subsequent layers decays and fails to propagate, whereas when the synaptic strength is too strong, the excessive propagation of firing rate occurs due to the burst firings. An optimal excitatory synaptic strength is found to best support the firing rate propagation. On the other hand, if both the excitatory and inhibitory neurons are considered in the feedforward neuronal network, the relative strength between inhibitory and excitatory synapses also plays a critical role in firing rate propagation. Better tuning of this factor can help feedforward neuronal network maintain better propagation performance [4].

3.3 Applications and Computations

Although the synfire propagation mode and the firing rate mode are quite different, the previous results demonstrated that a single network with different system parameters can support stable and robust signal propagation in both of the two modes, for example, they can be bridged by the background noise and synaptic strength [14], [18], [19]. Both the synfire mode and the firing rate mode can be used for applications and computations. Here we only give several typical examples but do not discuss their principle in detail. The synfire mode was widely used to recognize the translation-invariant pattern in a picture and parse the auditory scene [20]. Another important application for the synfire mode was proposed by Bienenstock, who used the synfire model for implementing the compositionality of cognitive functions [21]. The properties of the firing rate mode are also developed for computation. Van Rossum et al. implemented a local motion detector based on the firing rate mode and demonstrated that fast computations are feasible using rate-based models combined with population coding [14].

4 Signal Propagation in Feedforward Neuronal Networks with Unreliable Synapses

Neuron transmits spikes to postsynaptic neurons through synapses. Therefore, synapses can be regarded as the communication bridges between different neurons. It is known that the information processing in the brain is highly reliable. However, some biological experiments have demonstrated that the microscopic mechanism of synaptic transmission displays the unreliable property [1], [22]-[24]. Such unreliability is attributed to the probabilistic neurotransmitter release of the synaptic vesicles [25], [26]. For real biological neural systems, the successful spike transmission rates between 0.1 and 0.9 are widely reported in the literature [1], [27], [28]. Since the communication between real biological neurons indeed displays the unreliable property, a naturally arising question is how the unreliable synapses influence the signal propagation in feedforward neuronal networks. Here we first propose a more generalized modeling framework based on a heuristic unreliable synaptic model, then review and discuss several new computational results in our recent work.

Now we start to introduce the generalized modeling framework used to study the signal propagation in feedforward neuronal networks. In fact, our proposed generalized modeling framework is similar to the basic modeling framework. Here we also consider a feedforward neuronal coupled in an all-to-all fashion, and use the IF neuron model to mimics the action potential firing dynamics of biological neurons. The only difference between these two modeling frameworks lies in the synaptic interaction. In the generalized modeling framework, a stochastic on-off process is introduced to mimic the probabilistic transmitter release of the real biological synapses. When a presynaptic neuron fires a spike, we let the corresponding postsynaptic neurons receive it with a successful transmission probability p ($p<1$). The above stochastic on-off process is used to describe whether the neurotransmitter is successfully released or not (for detail, please see Ref. [13]).

In a very recent work, we have studied both the synfire propagation and firing rate propagation in the feedforward neuronal networks with unreliable synapses [13]. We found that the parameters of the unreliable synapses have significant effects on the performance of signal propagation. In the study of synfire mode, three types of synfire propagation can be observed depending on whether the synchronous spike packet can be successfully and stably transmitted to the final layer of the feedforward neuronal network. Stable synfire propagation only occurs for suitable combination of the successful transmission probability as well as the excitatory synaptic strength. Once parameters fall into the stable synfire propagation regime, a high synaptic reliability or a strong excitatory synaptic strength is able to support the synfire propagation in feedforward neuronal networks with better performance and faster transmission speed. In the study of firing rate propagation, our main finding is that there exists a suitable intermediate successful transmission probability to support the optimal firing rate propagation. In order to clarify the differences between the synaptic unreliability and network randomness, we make comparisons on the propagation performance between the feedforward neuronal network with unreliable synapses (unreliable model) and the corresponding feedforward neuronal network with random synaptic connections (random model). For the synfire mode, it is found that, compared to the random model, the unreliable model is able to suppress the occurrence of synfire instability propagation to a certain degree. While for the firing rate mode, the random model can better support the firing rate propagation in small successful transmission regime for strong excitatory synaptic strength. These results suggest that it is better not to simply use the random connections to replace the unreliable synapses in modeling research.

5 Conclusions and Discussions

Neuronal networks with feedforward structure provide us an effective way to examine the neural activity propagation through multiple brain regions. The study of signal propagation in multilayered feedforward neuronal networks can help us better understand the fundamental information transmission mechanism in the brain. So far, there have been many research results proposed, but few surveys in this filed. This paper introduced the basic modeling framework about the signal propagation in feedforwrd neuronal network, and two important propagation modes and their relevant results. Moreover, a more generalized modeling framework based on unreliable synapses was proposed and the effects of unreliable synapses were discussed. We believe that the results obtained based on our proposed unreliable synaptic models might be more realistic than those obtained based on the traditional reliable synaptic models. This is because the communication between biological neurons indeed displays the unreliable properties. We surmise that, in biological neural systems, neurons may make full use of the characteristics of unreliable synapses to transmit neural information in an adaptive way, that is, switching between different signal propagation modes freely as required [13]. We hope that this work will be useful and valuable for further research in this field. Due to the limitation of the page space, this paper only deals with signal propagation in isolated feedforward neuronal network. The case of signal propagation in embedded feedforward neuronal networks should be summarized in the further work.

Acknowledgments. This work is supposed by the award of the ongoing best PhD thesis support from the University of Electronic Science and Technology of China.

References

1. Gerstner, W., Kistler, W.M.: Spiking Neuron Models. Cambridge University Press, Cambridge (2002)
2. Dayan, P., Abbott, L.F.: Theoretical neuroscience: Computaional and mathematical modeling of neural systems. MIT Press, Cambridge (2001)
3. Vogels, P.T., Abbott, L.F.: Signal propagation and logic gating in networks of integrate-and-fire neurons. Journal of Neurosci. 25, 10786–10795 (2005)
4. Abeles, M.: Corticonics: Neural circuits of the cerebral cortex. Cambridge Uinversity Press, New York (1991)
5. Aertsen, A., Diesmann, M., Gewaltig, M.O.: Propagation of synchronous spiking activity in feedforward neural networks. Journal of Physiology-Paris 90, 243–247 (1996)
6. Diesmann, M., Gewaltig, M.O., Aertsen, A.: Stable propagation of synchronous spiking in cortical neural networks. Nature 402, 529–533 (1999)
7. Diesmann, M., Gewaltig, M.O., Rotter, S., Aertsen, A.: State space analysis of synchronous spiking in cortical neural networks. Neurocomputing 38-40, 565–571 (2001)
8. Diesmann, M.: Conditions for stable propagation of synchronous spiking in cortical neural networks: Single neuron dynamics and network properties. Ph.D. thesis, University of Bochum (2002)
9. Gewaltig, M.O., Diesmann, M., Aertsen, A.: Propagation of cortical synfire activity: Survival probability in single trials and stability in the mean. Neural Networks 14, 657–673 (2001)
10. Cateau, H., Fukai, T.: Fokker-Planck approach to the pulse packet propagation in synfire chain. Neural Networks 14, 675–685 (2001)
11. Kumar, A., Rotter, S., Aertsen, A.: Conditions for propagating synchronous spiking and asynchronous firing rates in a cortical network model. Journal of Neurosci. 28, 5268–5280 (2008)
12. Kumar, A., Rotter, S., Aertsen, A.: Spiking activity propagation in neuronal networks: Reconciling different perspectives on neural coding. Nature Reviews Neuroscience 11, 615–627 (2010)
13. Guo, D., Li, C.: Signal propagation in feedforward neuronal networks with unreliable synapses. J. Comput. Neurosci. (Epub ahead of print)
14. van Rossum, M.C.W., Turrigiano, G.G., Nelson, S.B.: Fast propagation of firing rates through layered networks of noisy neurons. Journal of Neurosci. 22, 1956–1966 (2002)
15. Reyes, A.D.: Synchrony-dependent propagation of firing rate in iteratively constructed networks in vitro. Nature Neuroscience 6, 593–599 (2003)
16. Vogels, T.P., Rajan, K., Abbott, L.F.: Neural Network Dynamics. Annu. Rev. Neurosci. 28, 357–376 (2005)
17. Wang, S.T., Wang, W., Liu, F.: Propagation of firing rate in a feed-forward neuronal network. Physical Review Letters 96, 018103 (2006)

18. Masuda, N., Aihara, K.: Bridging rate coding and temporal spike coding by effect of noise. Physical Review Letters 88, 248101 (2002)
19. Masuda, N., Aihara, K.: Duality of rate coding and temporal coding in multilayered feedforward networks. Neural Computation 15, 103–125 (2003)
20. Arnoldi, H.M., Englemeier, R.K., Brauer, W.: Invariant pattern recognition based on synfire chains. Biol. Cybern. 80, 433–447 (1999)
21. Bienenstock, E.: A model of neocortex. Network: Computation in Neural Systems 6, 179–224 (1995)
22. Raastad, M., Storm, J.F., Andersen, P.: Putative single quantum and single fiber excitatory postsynaptic currents show similar amplitude range and variability in rat hippocampal slices. European Journal of Neuroscience 4, 113–117 (1992)
23. Smetters, D.K., Zador, A.: Synaptic transmission: Noisy synapses and noisy neurons. Current Biology 6, 1217–1218 (1996)
24. Trommershäuser, J., Diesmann, M.: The effect of synaptic variability on the synchronization dynamics in feedforward cortical networks. Soc. Neurosci. Abstr. 64, 4 (2001)
25. Branco, T., Staras, K.: The probability of neurotransmitter release: Variability and feedback control at single synapses. Nature Reviews Neuroscience 10, 373–383 (2009)
26. Katz, B.: The release of neural transmitter substances. Liverpol University Press, Liverpool (1969)
27. Allen, C., Stevens, C.: An evaluation of causes for unreliability of synaptic transmission. Proc. Natl. Acad. Sci. USA 91, 10380–10383 (1995)
28. Stevens, C.F., Wang, Y.: Facilitation and depression at single central synapses. Neuron 14, 795–802 (1995)

Stability of Stochastic Cohen-Grossberg Neural Networks with Mixed Time Delay and Markovian Parameters

Junxiang Lu, ShanShan Wang, and Chengyi Zhang

School of Science, Xi'an Polytechnic University, No.19, Jinhua Rd(S),
Xi'an, P.R. China, PC:710048
{jun-xianglu}@163.com

Abstract. This paper mainly concerns stochastically asymptotical stability analysis problems for a class of stochastic Cohen-Grossberg neural networks with mixed time delays and Markovian parameters (SDCGNNswM). Based on an Lyapunov-Krasovskii functional and the stochastic stability analysis theory, a linear matrix inequality (LMI) approach is developed to derive the sufficient conditions guaranteeing the stochastically asymptotical stability of the equilibrium point. All the obtained results are presented in term of linear matrix inequalities. The efficiency of the proposed results is demonstrated via a numerical example.

Keywords: Cohen-Grossberg neural networks; Lyapunov function; Stochastic stability; Brownian motion.

1 Introduction

Since the Cohen-Grossberg neural networks was introduced by Cohen and Grossberg [1], this model has been widely studied due to their extensive applications in classification of patterns, associative memories, image processing, quadratic optimization, and other areas [2,3]. However, in electronic implementation of neural networks, there are inevitably some uncertainties due to the existence of modeling error, external disturbance and parameter fluctuation, which would lead to complex dynamical behaviors. Thus a good neural network should have certain stability against such uncertainties.

To the best our knowledge, a real system is usually affected by external perturbations which in many cases are of great uncertainty and hence may be treated as random, as pointed out by [4] that in real nervous systems synaptic transmission is a noisy process brought on by random fluctuations from the release of neurotransmitters, and other probability causes. Therefore, it is significant and of prime importance to consider stochastic effects to the stability property of neural networks (see [5,10]). In [5], Zidong Wang have investigated the asymptotical stability for stochastic Cohen-Grossberg neural networks with mixed time delays.

© Springer-Verlag Berlin Heidelberg 2011

Markovian jump systems introduced by [11], are the hybrid systems with two components in the state. The jump systems have the advantage of modeling the dynamic systems subject to abrupt variation in their structures, such as component failures or repairs, sudden environmental disturbance, changing subsystem interconnections, operating in different point of a nonlinear plant. The problem of stochastic robust stability for uncertain delayed neural networks with Markovian jumping parameters is investigated via linear matrix inequality technique in [12]. To the best our knowledge, only few works have been done on the stability analysis for Cohen-Grossberg neural networks with Markovian jumping parameters.

In this paper, we deal with the stochastic stability analysis problem for a class of SDCGNNswM. By utilizing a Lyapunov- Krasovskii function and conducting the stochastic analysis as well as LMIs, the criterion on stochastic stability and exponential stability are established. Note that LMIs can be easily solved by using the Matlab LMI toolbox. A numerical example is provided to show the usefulness of the proposed stability condition.

2 Problem Formulation

In this paper, the Cohen-Grossberg neural networks with mixed time delays and Markovian jumping parameters can be described by the following delay differential equations:

$$\begin{aligned} du_i(t) = &-a_i(u_i(t))[b_i(u_i(t)) - \sum_{j=1}^{n} a_{ij}(\eta_t) f_{1j}(u_j(t)) \\ &- \sum_{j=1}^{n} b_{ij}(\eta_t) f_{2j}(u_j(t-h)) - \sum_{j=1}^{n} c_{ij}(\eta_t) \int_{t-\tau}^{t} f_{3j}(u_j(s))ds + I_i(t)] \\ &+\psi(t, u_i(t), u_i(t-h))d\omega_i(t), \quad i = 1, 2, \dots, n, \end{aligned} \tag{1}$$

where $u_i(t)$ is the state of the ith unit at time t, $a_i(u_i(t))$ is the amplification function, $b_i(u_i(t))$ denotes the behave function, and $f_{kj}(u_i(t))(k = 1, 2, 3)$ are the activation functions. $a_{ij}(\eta_t)$, $b_{ij}(\eta_t)$, $c_{ij}(\eta_t)$ are the connection weight, the discretely delayed connection weight, the distributively delayed connection weight, respectively. $I_i(t)$ is external input at time t. The scalar $h > 0$ denotes the discrete time delay, whereas $\tau > 0$ denotes the distributed time delay. let $\sigma = \max\{h, \tau\}$. Moreover, $\omega(t) = (\omega_1(t), \omega_2(t), \dots, \omega_n(t))$ is n-dimensional Brownian motion defined on a complete probability space $(\Omega, \mathcal{F}, \mathcal{P})$ with a natural filtration $\{\mathcal{F}_t\}_{t\geq 0}$ generated by $\{\omega(s) : 0 \leq s \leq t\}$, where we associate Ω with the canonical space generated by $\omega(t)$, and denote by $\mathcal{F}$ the associated σ-algebra generated by $\omega(t)$ with the probability measure P. Let the random form process $\eta_t, t \in [0, +\infty)$ be a homogeneous, finite-state Markovian process with right continuous trajectories with generator $\Im = (\pi_{ij})$ and transition probability from mode i at time t to mode j at time $t + \delta$, $i, j \in S, (S = 1, 2, \dots, N)$:

$$p_{ij} = P(\eta_{t+\delta} = j | \eta_t = i) = \begin{cases} \pi_{ij}\delta + o(\delta), & \text{if } i \neq j, \\ 1 + \pi_{ij}\delta + o(\delta), & \text{if } i = j, \end{cases} \tag{2}$$

with transition probability rates $\pi_{ij} \geq 0$ for $i, j \in S, i \neq j$ and $\pi_{ii} = -\sum_{j=1,j\neq i}^{N} \pi_{ij}$, where $\delta > 0$ and $\lim_{\delta\to 0} o(\delta)/\delta = 0$. Note that the set S comprises the various operational modes of the system under study.

In this paper, the following assumptions are made on the amplification function, the behave function, the neuron activation function and Markovian process.

Assumption 1. For each $i \in 1, 2, \ldots, n$, the amplification function $ai(\cdot)$ is positive, bounded, and satisfies

$$0 < \underline{\alpha}_i \leq a_i(\cdot) \leq \bar{\alpha}_i, \tag{3}$$

where $\underline{\alpha}_i$ and $\bar{\alpha}_i$ are known positive constants.

Assumption 2. The behave function $b_i(x) : R \to R$ is continuous and differentiable, and

$$b_i'(x) \geq \gamma_i > 0 \quad \forall x \in R, \quad i = 1, 2, \ldots, n. \tag{4}$$

Assumption 3. The neuron activation function is bounded, and the following condition is satisfied:

$$|f_{kj}(x) - f_{kj}(y)| \leq l_{kj}^{f} |x - y|,\ \forall x, y \in R^n, \quad k = 1, 2, 3. \tag{5}$$

where l_{kj}^{f} is a constant.

Assumption 4. The mode η_t is available at time t.

Let $u(t, \phi)$ denote the state trajectory of the system (1)from the initial data $u(\theta) = \phi(\theta)$ on $-\sigma \leq \theta \leq 0$ in $L^2_{\mathcal{F},}([-\tau, 0]; R^n)$. Suppose that system (1) has an equilibrium point $u^* = [u_1^*, u_2^*, \ldots, u_n^*]^T$. For notation convenience, we shift the equilibrium point u^* to the origin by translating $x(t) = u(t) - u^*$ and write the system (1) into vector form, which yields the following system:

$$\begin{aligned} dx(t) = -\alpha(x(t))[\beta(x(t)) - A(\eta_t)g_1(x(t)) - B(\eta_t)g_2(x(t-h)) \\ -C(\eta_t)\textstyle\int_{t-\tau}^{t} g_3(x(s))ds] + \sigma(t, x(t), x(t-h))d\omega(t), \end{aligned} \tag{6}$$

where

$$\begin{aligned} & x(t) = (x_1(t), x_2(t), \ldots, x_n(t))^T, \qquad \omega(t) = (\omega_1(t), \omega_2(t), \ldots, \omega_n(t))^T \\ & \alpha(x(t)) = diag(\alpha_1(x_1(t)), \alpha_2(x_2(t)), \ldots, \alpha_n(x_2(t))), \\ & \beta(t) = (\beta_1(x_1(t)), \beta_2(x_2(t)), \ldots, \beta_n(x_n(t))) \\ & A(\eta_t) = (a_{ij}(\eta_t))_{n\times n}, B(\eta_t) = (b_{ij}(\eta_t))_{n\times n}, C(\eta_t) = (c_{ij}(\eta_t))_{n\times n}, \\ & g_k(x(t)) = (g_{k1}(x_1(t)), g_{k2}(x_2(t)), \ldots, g_{kn}(x_n(t)))^T, \quad k = 1, 2, 3, \\ & \alpha_i(x_i(t)) = a_i(x_i(t) + u_i^*), \qquad \beta_i(x_i(t)) = b_i(x_i(t) + u_i^*) - b_i(u_i^*), \\ & g_{kj}(x_j(t)) = f_{kj}(x_j(t) + u_j^*) - f_{kj}(u_j^*), \\ & \sigma(t, x_i(t), x_i(t-h)) = \psi(t, x_i(t) + u_i^*, xi(t-h) + u_i^*) - \psi(t, u_i^*, u_i^*). \end{aligned} \tag{7}$$

Assumption 5. The mapping $\sigma : R^+ \times R^n \times R^n \to R^n$ is globally Lipschitz continuous and satisfies the linear growth condition. Moreover,

$$\begin{aligned} & trace[\sigma^T(x(t), y(t), \eta_t)\sigma(x(t), y(t), \eta_t)] \leq x(t)^T \Sigma_1^T \Sigma_1 x(t) + y(t)^T \Sigma_2^T \Sigma_2 y(t), \\ & \sigma(t, 0) = 0. \end{aligned} \tag{8}$$

where $\Sigma_i, (i = 1, 2)$ are known constant matrix with n dimension.

It follows, respectively, from Assumption 1, Assumption 2, Assumption 3 that

$$\begin{aligned} &0 < \underline{\alpha}_i \le \alpha_i(\cdot) \le \bar{\alpha}_i, \\ &x_i(t)\beta_i(x_i(t)) \ge \gamma_i^2 x_i^2(t) > 0, \\ &|g_j(x)| \le l_j^f |x|, \quad i = 1, 2, \dots, n. \end{aligned} \tag{9}$$

In order to obtain our results, we need establishing the following definitions and lemmas:

Definition 1.(See [8]) The trivial solution of the SDCGNNswM is said to be stochastic asymptotically stable if for all initial state $\phi(0)$ and mode η_0 such that

$$\lim_{t\to\infty} E||x(t, \phi(0), \eta_0)|| = 0. \tag{10}$$

Lemma 2. (See [6]) Given any real matrices R_1, R_2 of appropriate dimensions such that $0 < R_3 = R_3^T$. Then, the following inequality holds:

$$2R_1^T R_2 \le R_1^T R_3 R_1 + R_2^T R_3^{-1} R_2. \tag{11}$$

Lemma 3. (Schur complement) Let $Q(x) = Q^T(x)$, $R(x) = R^T(x)$, $S(x)$ depend only on x and $R(x)$ is nonsingular, the following LMI

$$\begin{bmatrix} Q(x) & S(x) \\ S^T(x) & R(x) \end{bmatrix} > 0 \tag{12}$$

is equivalent to $R(x) > 0, Q(x) - S(x)R^{-1}(x)S(x)^T > 0$.

For simplicity, while $\eta_t = i$, the matrices $A(\eta_t), B(\eta_t), C(\eta_t)$ are presented by A_i, B_i, C_i.

Hence, we extract from (6), for $\eta_t = i \in S$, the following system

$$\begin{aligned} dx(t) = -\alpha(x(t))[&\beta(x(t)) - A_i g_1(x(t)) - B_i g_2(x(t-h)) \\ &-C_i \textstyle\int_{t-\tau}^t g_3(x(s))ds] + \sigma(t, x(t), x(t-h))d\omega(t), \end{aligned} \tag{13}$$

3 Main Results

In this section, we present a sufficient condition for stochastically asymptotic stability of the origin solution for the SDCGNNswM (13).

Theorem 1. Let the assumptions 1,2,3,4 and 5 be satisfied. Suppose that there exists symmetric and positive definite matrix $P_i, Q_{1i}, Q_{2i}, S_i, R_i, J_i$ such that the following LMI holds:

$$\Omega_1 = \begin{bmatrix} (1,1) & P_i\bar{\alpha}A_i & P_i\bar{\alpha}B_i & P_i\bar{\alpha}C_i \\ A_i^T\bar{\alpha}P_i & -R_i & 0 & 0 \\ B_i^T\bar{\alpha}P_i & 0 & -S_i & 0 \\ C_i^T\bar{\alpha}P_i & 0 & 0 & -J_i \end{bmatrix} < 0 \tag{14}$$

where

$$\begin{aligned}
&(1,1) = -\underline{\alpha} P_i \Gamma - \Gamma \underline{\alpha} P_i + L^f R_i L^f + \Sigma_1^T P_i \Sigma_1 + Q_{1i} + \tau Q_{2i} + \sum_{j=1}^{N} \pi_{ij} P_j, \\
&\Gamma = diag(\gamma_1, \gamma_2, \ldots, \gamma_n), \qquad Q_{1i} = L_f S_i L_f + \Sigma_2^T P_i \Sigma_2, \qquad Q_{2i} = \tau L_f J_i L_f \\
&L_f = (\max_{k=1,2,3}\{l_{k1}^f\}, \max_{k=1,2,3}\{l_{k2}^f\}, \cdots, \max_{k=1,2,3}\{l_{kn}^f\})^T.
\end{aligned} \tag{15}$$

Then the origin of system (13) is stochastically asymptotic stability.

Proof. Consider the following lyapunov functional:

$$V(t, x(t), i) = x^T(t) P_i x(t) + \int_{t-h}^{t} x^T(s) Q_{1i} x(s) ds + \int_{-\tau}^{0} \int_{t+s}^{t} x^T(v) Q_{2i} x(v) dv ds. \tag{16}$$

In this case, the infinitesimal generator of the Markov process $(t, x(t), i)$ becomes:

$$\begin{aligned}
LV(t, x(t), i) = &-2x^T(t) P_i \alpha(x(t))[\beta(x(t)) - A_i g_1(x(t)) - B_i g_2(x(t-h)) \\
&-C_i \textstyle\int_{t-\tau}^{t} g_3(x(s)) ds] + trace[\sigma^T(t,x) P_i \sigma(t,x)] \\
&+x^T(t) \sum_{j=1}^{N} \pi_{ij} P_j x(t) + x^T(t) Q_{1i} x(t) - x^T(t-h) Q_{1i} x(t-h) \\
&+\tau x^T(t) Q_{2i} x(t) - \textstyle\int_{t-\tau}^{t} x^T(s) Q_{2i} x(s) ds
\end{aligned} \tag{17}$$

Noticing that

$$\begin{aligned}
&-2x^T(t) P_i \alpha(x(t)) \beta(x(t)) \le -2\underline{\alpha} x^T(t) P_i \beta(x(t)) \\
&\quad = -2\underline{\alpha} \sum_{j=1}^{n} x_j(t) P_{ij} \beta_j(x_j(t) = -2\underline{\alpha} \sum_{j=1}^{n} P_{ij} x_j(t) \beta_j(x_j(t) \\
&\quad \le -2\underline{\alpha} \sum_{j=1}^{n} P_{ij} \gamma_j x_j^2(t) = -2\underline{\alpha} x^T(t) P_i \Gamma x(t)
\end{aligned} \tag{18}$$

where $\Gamma = diag(\gamma_1, \gamma_2, \ldots, \gamma_n)$.

$$\begin{aligned}
&2x^T(t) P_i \alpha(x(t)) A_i g_1(x(t)) \\
&\quad \le x^T(t) P_i \alpha(x(t)) A_i R_i^{-1} A_i^T \alpha(x(t)) P_i x(t) + g_1^T(x(t)) R_i g_1(x(t)) \\
&\quad \le x^T(t) P_i \bar{\alpha} A_i R_i^{-1} A_i^T \bar{\alpha} A_i P_i x(t) + x^T(t) L^f R_i L^f x(t),
\end{aligned} \tag{19}$$

$$\begin{aligned}
&2x^T(t) P_i \alpha(x(t)) B_i g_2(x(t-h)) \\
&\quad \le x^T(t) P_i \alpha(x(t)) B_i S_i^{-1} B_i^T \alpha(x(t)) P_i x(t) + g_2^T(x(t-h)) S_i g_2(x(t-h)) \\
&\quad \le x^T(t) P_i \bar{\alpha} B_i S_i^{-1} B_i^T \bar{\alpha} B_i P_i x(t) + x^T(t-h) L^f S_i L^f x(t-h),
\end{aligned} \tag{20}$$

$$\begin{aligned}
&2x^T(t) P_i \alpha(x(t)) C_i \textstyle\int_{t-\tau}^{t} g_3(x(s)) ds \\
&\quad \le x^T(t) P_i \alpha(x(t)) C_i J_i^{-1} C_i^T \alpha(x(t)) P_i x(t) \\
&\qquad + (\textstyle\int_{t-\tau}^{t} g_3(x(s)) ds)^T J_i \textstyle\int_{t-\tau}^{t} g_3(x(s)) ds \\
&\quad \le x^T(t) P_i \bar{\alpha} C_i J_i^{-1} C_i^T \bar{\alpha} C_i P_i x(t) + \tau \textstyle\int_{t-\tau}^{t} g_3^T(x(s)) ds x^T(t) J_i g_3(x(s)) ds, \\
&\quad \le x^T(t) P_i \bar{\alpha} C_i J_i^{-1} C_i^T \bar{\alpha} C_i P_i x(t) + \tau \textstyle\int_{t-\tau}^{t} x^T(s) L^f J_i L^f (x(s)) ds,
\end{aligned} \tag{21}$$

Let $Q_{2i} = \tau L_f J_i L_f, Q_{1i} = L_f S_i L_f + \Sigma_2^T P_i \Sigma_2$ and substituting (18-21) into (17), we have that

$$\begin{aligned}
LV(t,x(t),i) \leq & -2\bar{\alpha} x^T(t) P_i \bar{\alpha} x(t) + x^T(t) P_i \bar{\alpha} A_i R_i^{-1} A_i^T \bar{\alpha} A_i P_i x(t) \\
& + x^T(t) L^f R_i L^f x(t) + x^T(t) P_i \bar{\alpha} B_i S_i^{-1} B_i^T \bar{\alpha} B_i P_i x(t) \\
& + x^T(t-h) L^f S_i L^f x(t-h) + x^T(t) P_i \bar{\alpha} C_i J_i^{-1} C_i^T \bar{\alpha} C_i P_i x(t) \\
& + \tau \int_{t-\tau}^{t} x^T(s) L^f J_i L^f (x(s)) ds + x^T(t) Q_{1i} x(t) \\
& - x^T(t-h) Q_{1i} x(t-h) + \tau x^T(t) Q_{2i} x(t) \\
& + x^T(t) \Sigma_1^T P_i \Sigma_1 x(t) + x^T(t-h) \Sigma_2^T P_i \Sigma_2 x(t-h) \\
& + x^T(t) \sum_{j=1}^{N} \pi_{ij} P_j x(t) - \int_{t-\tau}^{t} x^T(s) Q_{2i} x(s) ds \\
= & \; x^T(t)[-\underline{\alpha} P_i \Gamma - \Gamma \underline{\alpha} P_i + L^f R_i L^f + \Sigma_1^T P_i \Sigma_1 + Q_{1i} + \tau Q_{2i} \\
& + \sum_{j=1}^{N} \pi_{ij} P_j + P_i \bar{\alpha} A_i R_i^{-1} A_i^T \bar{\alpha} A_i P_i + P_i \bar{\alpha} B_i S_i^{-1} B_i^T \bar{\alpha} B_i P_i \\
& + P_i \bar{\alpha} C_i J_i^{-1} C_i^T \bar{\alpha} C_i P_i] x(t)
\end{aligned} \tag{22}$$

From (14), we conclude that, for each mode $i, \Omega_1 < 0$. Therefore, if this inequality holds, it follows that $LV(t,x(t),i) < 0$. Thus the origin of system (13) is globally stochastically asymptotic stability. This completes the proof of Theorem 1.

In what follows, we will show that our results can be specialized to several cases including those have been studied extensively in the literature. All the corollaries given below are easy consequences of Theorem 1, hence the proofs are omitted.

We first consider the following Cohen-Grossberg neural network without stochastic perturbation:

$$\begin{aligned}
dx(t) = & -\alpha(x(t))[\beta(x(t)) - A_i g_1(x(t)) - B_i g_2(x(t-h)) \\
& - C_i \int_{t-\tau}^{t} g_3(x(s)) ds],
\end{aligned} \tag{23}$$

Corollary 2. Let the assumptions 1,2,3 be satisfied. Suppose that there exists symmetric and positive definite matrix $P_i, Q_{1i}, Q_{2i}, S_i, R_i, J_i$ such that the following LMI holds:

$$\Omega_3 = \begin{bmatrix} (1,1) & P_i \bar{\alpha} A_i & P_i \bar{\alpha} B_i & P_i \bar{\alpha} C_i \\ A_i^T \bar{\alpha} P_i & -R_i & 0 & 0 \\ B_i^T \bar{\alpha} P_i & 0 & -S_i & 0 \\ C_i^T \bar{\alpha} P_i & 0 & 0 & -J_i \end{bmatrix} < 0 \tag{24}$$

where

$$\begin{aligned}
& (1,1) = -\underline{\alpha} P_i \Gamma - \Gamma \underline{\alpha} P_i + L^f R_i L^f + Q_{1i} + \tau Q_{2i} + \sum_{j=1}^{N} \pi_{ij} P_j, \\
& \Gamma = diag(\gamma_1, \gamma_2, \ldots, \gamma_n), \qquad Q_{1i} = L_f S_i L_f, \qquad Q_{2i} = \tau L_f J_i L_f.
\end{aligned} \tag{25}$$

Then the origin of system (23) is asymptotic stability.

Remark 3. Although there have been some papers published on the stability analysis problems for Cohen-Grossberg neural network with mixed time delays

[13], to the best of the authors knowledge, there are few results concerning the simultaneous presence of discrete or distributed time delays and Markovian parameters. Hence, the results in Corollary 2 are still new.

If we are only interested in stochastic perturbation, the Cohen-Grossberg neural network (15) can be further reduced to

$$\begin{aligned} dx(t) = -\alpha(x(t))[\beta(x(t)) - Ag_1(x(t)) - Bg_2(x(t-h)) \\ -C\int_{t-\tau}^{t} g_3(x(s))ds] + \sigma(t, x(t), x(t-h))d\omega(t), \end{aligned} \tag{26}$$

Corollary 4. Let the assumptions 1,2,3,5 be satisfied. Suppose that there exists symmetric and positive definite matrix P, Q_1, Q_2, S, R, J such that the following LMI holds:

$$\Omega_5 = \begin{bmatrix} (1,1) & P\bar{\alpha}A & P\bar{\alpha}B & P\bar{\alpha}C \\ A^T\bar{\alpha}P & -R & 0 & 0 \\ B^T\bar{\alpha}P & 0 & -S & 0 \\ C^T\bar{\alpha}P & 0 & 0 & -J \end{bmatrix} < 0 \tag{27}$$

where

$$\begin{aligned} &(1,1) = -\underline{\alpha}P\Gamma - \Gamma\underline{\alpha}P + L^fRL^f + \Sigma_1^TP\Sigma_1 + Q_1 + \tau Q_2, \\ &\Gamma = diag(\gamma_1, \gamma_2, \dots, \gamma_n), \quad Q_1 = L_fSL_f + \Sigma_2^TP\Sigma_2, \quad Q_2 = \tau L_fJL_f. \end{aligned} \tag{28}$$

Then the origin of system (26) is stochastically asymptotic stability.

Remark 5. The Cohen-Grossberg neural network with mixed time delays (26) has been well investigated in [5]. The results in Corollary 4 alternative criteria based on LMIs approach.

If there appears only discrete time delay, the Cohen-Grossberg neural network (13) can be simplified to

$$dx(t) = -\alpha(x(t))[\beta(x(t)) - A_ig_1(x(t)) - B_ig_2(x(t-h))] + \sigma(t, x(t), x(t-h))d\omega(t), \tag{29}$$

Corollary 6. Let the assumptions 1,2,3,4 be satisfied. Suppose that there exists symmetric and positive definite matrix $P_i, Q_{1i}, S_i, R_i, J_i$ such that the following LMI holds:

$$\Omega_7 = \begin{bmatrix} (1,1) & P_i\bar{\alpha}A_i & P_i\bar{\alpha}B_i \\ A_i^T\bar{\alpha}P_i & -R_i & 0 \\ B_i^T\bar{\alpha}P_i & 0 & -S_i \end{bmatrix} < 0 \tag{30}$$

where

$$\begin{aligned} &(1,1) = -\underline{\alpha}P_i\Gamma - \Gamma\underline{\alpha}P_i + L^fR_iL^f + \Sigma_1^TP_i\Sigma_1 + Q_{1i} + \sum_{j=1}^{N}\pi_{ij}P_j, \\ &\Gamma = diag(\gamma_1, \gamma_2, \dots, \gamma_n), \quad Q_{1i} = L_fS_iL_f + \Sigma_2^TP_i\Sigma_2, \end{aligned} \tag{31}$$

Then the origin of system (29) is stochastically asymptotic stability.

Remark 7. Although in [13,14], the stability criteria of Cohen-Grossberg neural network with discrete time have been established in terms of LMIs, there are few results concerning stochastic perturbation and Markovian parameters. Hence, the results in Corollary 6 are still new.

4 An Example

In this section, we present a numerical example to illustrate our results.

Example. We consider the SDNNswM (1) with $n = 2$. Let the Markovian process governing the mode switching has generator

$$\Lambda = \begin{bmatrix} -0.4 & 0.4 \\ 0.3 & -0.3 \end{bmatrix}. \tag{32}$$

For the two operating modes, the associated data are:

$$A_1 = \begin{bmatrix} 0.3 & -1.8 \\ -1.1 & 1.6 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.6 \end{bmatrix}, \quad C_1 = \begin{bmatrix} 0.5 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}, \tag{33}$$

$$A_2 = \begin{bmatrix} 0.4 & -1.6 \\ -0.9 & 1.8 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0.6 & 0.7 \\ 1.1 & -1.2 \end{bmatrix}, \quad C_2 = \begin{bmatrix} 0.7 & -0.3 \\ -1.1 & 0.5 \end{bmatrix} \tag{34}$$

and

$$L_f = 0.2I, \bar{\alpha} = 0.8, \underline{\alpha} = 0.7, \Sigma_1 = \Sigma_2 = 0.08I, \tau = 0.5, h = 0.12, \tag{35}$$

where I is identity matrix.

Then using the MATLAB LMI toolbox, we can obtain the following feasible solution for LMIs (14):

$$\begin{aligned}
P_1 &= \begin{bmatrix} 0.6579 & 0.2209 \\ 0.2209 & 0.7027 \end{bmatrix}, & Q_{11} &= \begin{bmatrix} 0.8280 & 0.2505 \\ 0.2505 & 0.7767 \end{bmatrix}, & Q_{21} &= \begin{bmatrix} 0.1854 & 0.0474 \\ 0.0474 & 0.1969 \end{bmatrix}, \\
R_1 &= \begin{bmatrix} 1.3342 & -0.5533 \\ -0.5533 & 2.2818 \end{bmatrix}, & S_1 &= \begin{bmatrix} 0.9462 & 0.2532 \\ 0.2532 & 0.8400 \end{bmatrix}, & J_1 &= \begin{bmatrix} 0.9269 & 0.2370 \\ 0.2370 & 0.9845 \end{bmatrix}, \\
P_2 &= \begin{bmatrix} 0.7910 & 0.1993 \\ 0.1993 & 0.4233 \end{bmatrix}, & Q_{21} &= \begin{bmatrix} 0.9712 & 0.2122 \\ 0.2122 & 0.5487 \end{bmatrix}, & Q_{22} &= \begin{bmatrix} 0.2117 & 0.0291 \\ 0.0291 & 0.1483 \end{bmatrix}, \\
R_2 &= \begin{bmatrix} 1.1107 & -0.3879 \\ -0.3879 & 2.1193 \end{bmatrix}, & S_2 &= \begin{bmatrix} 1.2328 & 0.0617 \\ 0.0617 & 0.7858 \end{bmatrix}, & J_2 &= \begin{bmatrix} 1.0538 & 0.1453 \\ 0.1453 & 0.7413 \end{bmatrix},
\end{aligned} \tag{36}$$

Then the conditions of Theorem 1 are satisfied. Therefore, the origin solution of SDNNswM (1) is stochastically asymptotical stability.

Acknowledgments. This work was supported by by Shanxi Province Science Foundation for Youths,China(No. 2010JQ1016), the Science Research Foundation of Shaanxi Province Department of Education, China(No. 2010JK560).

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 Cybernetics SMC 13, 815–821 (1983)
2. Ye, H., Michel, A.N., Wang, K.: Qualitive analysis of Cohen-Grossberg neural networks with multiple delays. Phys. Rev. E 51, 2611–2618 (1995)

3. Grossberg, S.: Nonlinear neural networks: principles, mechanisms, and archi- tectures. Neural Networks 1, 17–61 (1988)
4. Haykin, S.: Neural Networks. Prentice-Hall, Englewood Cliffs (1994)
5. Wang, Z., et al.: Stability analysis of stochastic Cohen-Grossberg neural net works with mix delays. IEEE Transaction on Neural Networks 17, 814–820 (2006)
6. Wan, L., Sun, J.: Mean square exponential stability of stochastic delayed Hopfield neural networks. Physics Letters A 343, 300–318 (2006)
7. Wang, Z., et al.: Robust stability for stochastic delay neural networks with time delays. Nonlinear Analysis: Real world Applications 7, 1119–1128 (2006)
8. Blythe, S., Mao, X., Liao, X.: Stability of stochatic delay neural networks. Journal of the Franklin Institute 338, 481–495 (2001)
9. Liao, X., Mao, X.: Exponential stability and instability of stochastic neural networks. Stochast. Anal. Appl. 14(2), 165–185 (1996)
10. Lu, J.-X., Ma, Y.-C.: Mean square exponential stability and periodic solutions of stochastic delay cellular neural networks. Chaos, Solitons and Fractals 38, 1323–1331 (2008)
11. Krasovskii, N.M., Lidskii, E.A.: Analytical design of controllers in systems with random attributes. Autom. Remote Control 22, 1021–2025 (1961)
12. Xie, L.: Srochastic roust stability analysis for Markovian jumping neural net- works with time delays. Proceedings IEEE, Networking, Sensing and Control 19-22, 923–928 (2005)
13. Arik, S.: Global stability analysis of Cohen-Crossberg neural networks with time varying delays. Phys. Lett. A 341, 410–421 (2005)
14. Rong, L.: LMI-based critera for robust stability of Cohen-Crossberg neural networks with delay. Phy. Lett. A 339, 63–73 (2005)

Exponential Stability of Stochastic Neural Networks with Mixed Time-Delays*

Xuejing Meng[1,**], Maosheng Tian[2], Peng Hu[1], and Shigeng Hu[1]

[1] School of Mathematics and Statistics, Huazhong University of Science and Technology, Wuhan 430074, P.R. China
[2] State Engineering Research Center of Numerical Control System of Huazhong University of Science and Technology, Wuhan 430074, P.R. China
mengxuejing@gmail.com
http://www.springer.com/lncs

Abstract. This paper investigates the exponential stability of stochastic neural networks with unbounded discrete delays and infinitely distributed delays. By using Lyapunov functions, the semi-martingale convergence theorem and some inequality techniques, the exponential stability in mean square and almost sure exponential stability are obtained. To overcome the difficulties from unbounded delays, some new techniques are introduced. Some earlier results are improved and generalized. An example is given to illustrate the results.

Keywords: Stochastic neural networks; Unbounded delay; Distributed delay; Stability.

1 Introduction

In recent years, there have been an increasing research interest in the study of stochastic neural networks, see, for example, [3,6,7,11] and the references therein. Raska et al. [12] introduced a cellular neural network with discrete delays to deal with motion-related signal processing problems. In real neural networks, there usually have a spatial extent due to the presence of an amount of parallel pathways with a variety of axon sizes and lengths [2]. Therefore, there will be a distribution of conduction velocities along these pathways. Tank and Hopfield [14] proposed a neural circuit with distributed delays described by intergo-differential equations for solving a general problem of recognizing patterns in a time-dependent signal. To date, many researchers studied the stochastic neural networks with discrete delays and distributed delays (c.f. [1,5,8,9,15]).

However, the discrete time delays they discussed are bounded in the papers mentioned above. In delayed neural networks, in most situations, delays are

* This research was supported by the fundamental research funds for the central universities under grant No.2010MS130.
** Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

variable, and in fact unbounded [16]. In this paper, we study a generalized stochastic neural networks as follows:

$$dx(t) = \Big[-Bx(t)+AG(t,y(t))+\int_{-\infty}^{0} Dx(t+\theta)d\mu(\theta)\Big]dt+\sigma(t,x(t),y(t),x_t)dw(t), \tag{1}$$

where $x(t) \in \mathbb{R}^n$, $y(t) = (y_1(t),\ldots,y_n(t))^{\mathrm{T}}$, $y_i(t) = x_i(t-\delta_i(t))$, $\delta_i(t)$ is delay function which may be unbounded. $x_t = \{x(t+\theta) : -\infty < \theta \le 0\}$ is regarded as a $C_b((-\infty,0],\mathbb{R}^n)$-valued stochastic process. $B = \mathrm{diag}(b_i)$ with $b_i > 0$. $A = (a_{ij}) \in \mathbb{R}^{n\times n}$, $D = (d_{ij}) \in \mathbb{R}^{n\times n}$. Both functions $G(t,y) : \mathbb{R}_+ \times \mathbb{R}^n \to \mathbb{R}^{n\times n}$ and $\sigma(t,x,y,\varphi) : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^n \times C_b \to \mathbb{R}^{n\times m}$ are Borel measurable and locally Lipschitz continuous.

If $0 \le \delta_i(t) \le \tau < \infty$ $(1 \le i \le n)$, Eq.(1) covers the system which studied in [1] as a special case. If $0 \le \delta_i(t) \le \tau < \infty$ and remove the distributed delays, Eq.(1) becomes

$$dx(t) = [-Bx(t) + AG(t,y(t))]dt + \sigma(t,x(t),y(t))dw(t), \tag{2}$$

which is studied by Huang et al. in [4,13].

Stability is a basic knowledge for dynamical systems and is useful in application to the real life system. The main aim of this paper is to study the exponential stability in mean square and almost sure exponential stability. The main results are provided in Section 3. We also give an example to illustrate the results.

2 Preliminaries

Notations. Denote by $C_b = C_b((-\infty,0],\mathbb{R}^n)$ the family of all bounded continuous functions φ from $(-\infty,0]$ to $\mathbb{R}^n$ with the norm $\|\varphi\| = \sup_{-\infty<\theta\le 0}|\varphi(\theta)|$, which forms a Banach space. Let $\lambda_M(H)$ denote the biggest eigenvalue of matrix H. Let $w(t)$ be an m-dimensional Brownian motion defined on a complete probability space $(\Omega,\mathcal{F},\mathbb{P})$ with a nature filtration $\{\mathcal{F}_t\}_{t\ge 0}$. In this paper, *const* always represents some positive constant whose precise value is not important.

For any $V(x) \in C^2(\mathbb{R}^n;\mathbb{R}_+)$ and $t \ge 0$, $x,y \in \mathbb{R}^n$, $\varphi \in C_b$, define

$$\begin{aligned}\mathcal{L}V(t,x,y,\varphi) = V_x(x)\Big[-Bx + AG(t,y) + \textstyle\int_{-\infty}^{0} D\varphi(\theta)d\mu(\theta)\Big]\\ +\frac{1}{2}\mathrm{trace}[\sigma^{\mathrm{T}}(t,x,y,\varphi)V_{xx}(x)\sigma(t,x,y,\varphi)].\end{aligned} \tag{3}$$

If $x(t)$ is a solution of Eq.(1), applying the Itô formula to $V(x(t))$ yields

$$dV(x(t)) = LV(x(t))dt + V_x(x(t))\sigma(t,x(t),y(t),x_t)dw(t),$$

where $LV(x(t)) = \mathcal{L}V(t,x(t),y(t),x_t)$.

Let $\Delta_i(t) = t - \delta_i(t)$, $\delta_i(t) \in C^1(\mathbb{R}_+;\mathbb{R}_+)$ and assume

$$\eta_i := \inf_{t\ge 0} \Delta_i'(t) > 0, \quad (1 \le i \le n) \tag{4}$$

which clearly shows that $\Delta_i(t)$ is a strictly increasing function on $[0, \infty)$ and has the inverse function $\Delta_i^{-1}(s)$ defined on $[-\delta_i(0), \infty)$ with the following property

$$[\Delta_i^{-1}(s)]' = \frac{1}{\Delta_i'(t)} \leq \eta_i^{-1}. \quad (s = \Delta_i(t)) \tag{5}$$

It is easy to obtain the following lemma.

Lemma 1. *Let η_i be defined by (4), then $\eta_i \leq 1$.*

For any $\alpha, \beta \geq 0$ and $\varphi \in C_b$, define $\mathcal{T}_{\alpha\beta}(\varphi) = \int_{-\infty}^{0} e^{\alpha\theta}|\varphi(\theta)|^{\beta} d\theta$ and $C(\alpha, \beta) = \{\varphi \in C_b : \mathcal{T}_{\alpha\beta}(\varphi) < \infty\}$. Denote by M_0 the family of all probability measures on $(-\infty, 0]$. For any $\mu \in M_0$ and $\varepsilon \geq 0$, define

$$\begin{cases} \mu_\varepsilon := \int_{-\infty}^{0} e^{-\alpha\theta} d\mu(\theta); \\ M_\varepsilon = \{\mu \in M_0 : \mu_\varepsilon < \infty\}. \end{cases} \tag{6}$$

This paper often uses the following function Φ_ε:

$$\Phi_\varepsilon := \Phi_\varepsilon(t, x, y, \varphi) = \sum_{i=1}^{n} K_i \left(\int_{-\infty}^{0} \varphi_i^2(\theta) d\mu(\theta) - \mu_\varepsilon x_i^2 \right) + \sum_{i=1}^{n} L_i \left(e^{-\varepsilon\delta_i(t)} y_i^2 - \eta_i^{-1} x_i^2 \right), \tag{7}$$

for any $t \geq 0$, $x, y \in \mathbb{R}^n$ and $\varphi \in C_b$, where μ_ε is defined by (6), $\varepsilon > 0$ and K_i, L_i $(1 \leq i \leq n)$ are nonnegative constants.

Lemma 2. *Let Φ_ε be defined by (7) and $0 \leq q \leq \varepsilon$. If $x(t) = x(t, \xi)(-\infty < t < \sigma)$ is a solution to Eq.(1) with initial data $\xi \in C(q, 2)$, then*

$$\int_0^t e^{qs} \Phi_\varepsilon(s, x(s), y(s), x_s) ds \leq \text{ const.} \quad (0 \leq t < \sigma).$$

Proof. By the Fubini theorem and a substitution technique, the result can be easily derived. □

3 Stability Results

This section aims to establish the stability results for Eq.(1).

Theorem 1. *Assume that there exist positive constants ε and m_i $(1 \leq i \leq n)$ such that the function $V(x) = |x|^2$ satisfies*

$$\mathcal{L}V(t, x, y, \varphi) \leq \Phi_\varepsilon - \sum_{i=1}^{n} m_i x_i^2 \tag{8}$$

for any $t \geq 0$, $x, y \in \mathbb{R}^n$ and $\varphi \in C_b$, where Φ_ε is defined by (7). Then there exists $q \in (0, \varepsilon]$, for any given $\xi \in C(q, 2)$, Eq.(1) admits a unique global solution $x(t, \xi)$ which satisfies

$$\limsup_{t \to \infty} t^{-1} \ln(\mathbb{E}|x(t, \xi)|^2) \leq -q, \tag{9}$$

$$\limsup_{t \to \infty} t^{-1} \ln |x(t, \xi)| \leq -q/2, \qquad a.s.. \tag{10}$$

Proof. The proof will be divided into two steps.

Step 1. Existence of the global solution Fix $\xi \in C(q, 2)$, there exists a unique maximal local solution $x(t) = x(t, \xi)$ for $-\infty < t < \sigma$ to Eq.(1), where σ is the explosion time. Let k_0 be a sufficiently large positive number such that $\|\xi\| \leq k_0$. For each integer $k \geq k_0$, define the stopping time

$$\sigma_k = \inf\{0 \leq t < \sigma : V(x(t)) \geq k\}.$$

Clearly, σ_k is increasing and so $\sigma_k \to \sigma_\infty \leq \sigma$ as $k \to \infty$. If we can show $\sigma_\infty = \infty$, a.s., then $\sigma = \infty$ a.s., which implies the desired result. This is also equivalent to proving that, for any $t > 0$, $\mathbb{P}(\sigma_k \leq t) \to 0$ as $k \to \infty$. Letting $t_k = t \wedge \sigma_k$, by condition (8), applying the Itô formula and Lemma 2 yields

$$\begin{aligned} k\mathbb{P}(\sigma_k \leq t) = \mathbb{E}[I_{\{\sigma_k \leq t\}} V(x(t_k))] &\leq \mathbb{E}V(x(t_k)) \leq const + \mathbb{E}\int_0^{t_k} LV(x(s))ds \\ &\leq const + \mathbb{E}\int_0^{t_k} \left[\Phi_\varepsilon(s, x(s), y(s), x_s) - \sum_{i=1}^n m_i |x_i(s)|^2\right] ds \\ &\leq const - \sum_{i=1}^n m_i \mathbb{E}\int_0^{t_k} |x_i(s)|^2 ds \\ &\leq const. \end{aligned}$$

This implies $\mathbb{P}(\sigma_k \leq t) \to 0$ as $k \to \infty$, as required.

Step 2. Stability Choose $q > 0$ and q is sufficiently small. Fix $\xi \in C(q, 2)$. Let $h(t) = e^{qt} V(x(t))$. Applying the Itô formula, we have

$$h(t) = h(0) + I + M(t),$$

where $M(t) = \int_0^t e^{qs} V_x(x(s))\sigma(s, x(s), y(s), x_s)dw(s)$ is a continuous local martingale with $M(0) = 0$, and

$$\begin{aligned} I &= \int_0^t e^{qs}[LV(x(s)) + qV(x(s))]ds \\ &\leq \int_0^t e^{qs}\left[\Phi_\varepsilon(s, x(s), y(s), x_s) - \sum_{i=1}^n m_i |x_i(s)|^2 + qV(x(s))\right] ds \\ &\leq const. \end{aligned}$$

Applying the semi-martingale convergence theorem (see [10], P.44, Theorem 7.4) gives

$$\limsup_{t\to\infty} \mathbb{E}e^{qt} V(x(t)) < \infty, \quad \limsup_{t\to\infty} e^{qt} V(x(t)) < \infty. \ \ a.s..$$

This directly implies assertions (9) and (10), as required. □

Next, we will apply Theorem 1 to get some more applicable criteria. Before giving the main results, some conditions and notations will be presented firstly.

For any $t \geq 0$, $x, y \in \mathbb{R}^n$ and $\varphi \in C_b$, assume that $G_i(t,y)$ and $\sigma(t,x,y,\varphi)$ satisfy

$$|G_i(t,y)| \leq \beta_i |y_i| e^{-\varepsilon\delta_i(t)}, \tag{11}$$

$$|\sigma(t,x,y,\varphi)|^2 \leq \sum_{i=1}^{n} \Big[\lambda_i x_i^2 + \tilde{\lambda}_i \int_{-\infty}^{0} \varphi_i^2(\theta) d\mu(\theta) + \bar{\lambda}_i y_i^2 e^{-\varepsilon\delta_i(t)} + \mu_i G_i^2(t,y)\Big] \tag{12}$$

where $\mu \in M_\varepsilon$. β_i, λ_i, $\tilde{\lambda}_i$, $\bar{\lambda}_i$ and μ_i $(1 \leq i \leq n)$ are nonnegative constants.

For the sake of simplicity, we introduce the following notations:

$$\tilde{A} = (\tilde{a}_{ij})_{n\times n}, \quad \tilde{a}_{ij} = |a_{ij}|\beta_j, \quad q_i = \tilde{q}_i + \bar{q}_i \eta_i^{-1}, \quad s_i = \mu_i \beta_i^2 \eta_i^{-1} \tag{13}$$

$$\hat{\lambda}_i = \lambda_i + \tilde{\lambda}_i + \bar{\lambda}_i \eta_i^{-1}, \qquad \hat{\rho}_i = \rho_i + \tilde{\rho}_i + \bar{\rho}_i \eta_i^{-1} \tag{14}$$

Theorem 2. *Under conditions (11) and (12), if there exist constants $\tilde{q}_i$, $\bar{q}_i$, ρ_i, $\bar{\rho}_i$, $\tilde{\rho}_i$ $(1 \leq i \leq n)$ such that the following conditions are satisfied:*

$$\tilde{\rho}_i \leq \tilde{q}_i, \quad \bar{q}_i \geq \bar{\rho}_i, \tag{15}$$

$$\hat{\lambda}_i + s_i < \hat{\rho}_i, \tag{16}$$

$$\begin{pmatrix} x^{\mathrm{T}} & z^{\mathrm{T}} & y^{\mathrm{T}} \end{pmatrix} H \begin{pmatrix} x \\ z \\ y \end{pmatrix} \leq -\sum_{i=1}^{n} (\rho_i x_i^2 + \tilde{\rho}_i z_i^2 + \bar{\rho}_i y_i^2), \tag{17}$$

where

$$H = \begin{pmatrix} diag(q_i - 2b_i) & D & \tilde{A} \\ D^{\mathrm{T}} & -diag(\tilde{q}_i) & 0 \\ \tilde{A}^{\mathrm{T}} & 0 & -diag(\bar{q}_i) \end{pmatrix}, \tag{18}$$

then there exists $q > 0$, for any given initial data $\xi \in C(q,2)$, Eq.(1) admits a unique global solution $x(t,\xi)$ which satisfies (9) and (10).

Proof. For $t \geq 0$, $x, y \in \mathbb{R}^n$ and $\varphi \in C_b$, let $V(x) = |x|^2$ and $z = \int_{-\infty}^{0} \varphi(\theta) d\mu(\theta)$, applying the Itô formula to $V(x)$ and using conditions (11) (12) and (15)-(17) give

$$\begin{aligned}
& \mathcal{L}V(t,x,y,\varphi) \\
= \; & 2x^{\mathrm{T}}(-Bx + AG(t,y) + Dz) + |\sigma(t,x,y,\varphi)|^2 \\
= \; & \begin{pmatrix} x^{\mathrm{T}} & z^{\mathrm{T}} & y^{\mathrm{T}} \end{pmatrix} H \begin{pmatrix} x \\ z \\ y \end{pmatrix} + \sum_{i=1}^{n} (\tilde{q}_i z_i^2 - q_i x_i^2 + \bar{q}_i y_i^2) + 2x^{\mathrm{T}} AG(t,y) \\
& -2x^{\mathrm{T}} \tilde{A} y + |\sigma(t,x,y,\varphi)|^2 \\
\leq \; & -\sum_{i=1}^{n} (\rho_i x_i^2 + \tilde{\rho}_i z_i^2 + \bar{\rho}_i y_i^2) + \sum_{i=1}^{n} (\tilde{q}_i z_i^2 - q_i x_i^2 + \bar{q}_i y_i^2) \\
& + \sum_{i=1}^{n} \Big[\lambda_i x_i^2 + \tilde{\lambda}_i \int_{-\infty}^{0} \varphi_i^2(\theta) d\mu(\theta) + \bar{\lambda}_i y_i^2 e^{-\varepsilon\delta_i(t)} + \mu_i G_i^2(t,y)\Big]
\end{aligned}$$

$$\begin{aligned}
&\le \sum_{i=1}^{n}(\lambda_i-\rho_i-q_i)x_i^2+\sum_{i=1}^{n}(\tilde{\lambda}_i-\tilde{\rho}_i+\tilde{q}_i)\int_{-\infty}^{0}\varphi_i^2(\theta)d\mu(\theta)\\
&\quad+\sum_{i=1}^{n}(\bar{\lambda}_i-\bar{\rho}_i+\bar{q}_i+\mu_i\beta_i^2)y_i^2e^{-\varepsilon\delta_i(t)}\\
&:=\varPhi_\varepsilon-\sum_{i=1}^{n}m_ix_i^2,
\end{aligned}$$

where

$$\begin{aligned}
\varPhi_\varepsilon=&\sum_{i=1}^{n}(\tilde{\lambda}_i-\tilde{\rho}_i+\tilde{q}_i)\Big[\int_{-\infty}^{0}\varphi_i^2(\theta)d\mu(\theta)-\mu_\varepsilon x_i^2\Big]\\
&+\sum_{i=1}^{n}(\bar{\lambda}_i-\bar{\rho}_i+\bar{q}_i+\mu_i\beta_i^2)\big[y_i^2e^{-\varepsilon\delta_i(t)}-\eta_i^{-1}x_i^2\big]
\end{aligned}$$

is a function in the form of (7), and

$$\begin{aligned}
m_i|_{\varepsilon=0}&=\rho_i+q_i-\lambda_i-\tilde{\lambda}_i+\tilde{\rho}_i-\tilde{q}_i-(\bar{\lambda}_i-\bar{\rho}_i+\bar{q}_i+\mu_i\beta_i^2)\eta_i^{-1}\\
&\ge\hat{\rho}_i-\hat{\lambda}_i-s_i>0.
\end{aligned}$$

This shows condition (8) is satisfied. Applying Theorem 1 gives the desired results. □

Choose $\rho_i=\bar{\rho}_i=\tilde{\rho}_i\equiv\rho$, we can derive the following result.

Corollary 1. *Under conditions (11) and (12), if there exist constants $\tilde{q}_i$, $\bar{q}_i$ $(1\le i\le n)$ and ρ such that $\lambda_M(H)\le-\rho$, and moreover,*

$$\rho\le\tilde{q}_i\wedge\bar{q}_i, \tag{19}$$

$$\hat{\lambda}_i+s_i<\rho(2+\eta_i^{-1}),(1\le i\le n), \tag{20}$$

where H is defined by (18), then the conclusions of Theorem 2 hold.

Corollary 2. *Under conditions (11) and (12), if*

$$2b_i>\sum_{j=1}^{n}|d_{ij}|+\sum_{j=1}^{n}|\tilde{a}_{ij}|+\sum_{j=1}^{n}|d_{ji}|+\sum_{j=1}^{n}|\tilde{a}_{ji}|\eta_i^{-1}+s_i+\hat{\lambda}_i, \tag{21}$$

then the conclusions of Theorem 2 hold.

Proof. Choosing $\tilde{q}_i=\bar{q}_i=0$ and using (21) we can testify condition (15)-(17) in Theorem 2. Applying Theorem 2 gets the desired results. Here we omit its process. □

Remark 1. The application of Theorem 1 and Corollary 1 need choosing some parameters such as $\tilde{q}_i$, $\bar{q}_i$, etc.. Corollary 2 is described only in terms of the system parameters hence it can be verified easily.

In the following, we consider a simple case. Simplify condition (12) and assume that

$$|\sigma(t,x,y,\varphi)|^2 \le \sum_{i=1}^{n} \mu_i G_i^2(t,y). \tag{22}$$

Corollary 3. *Under conditions (11) and (22), if there exist constants $\tilde{q}_i$, $\bar{q}_i$ $(1 \le i \le n)$ and ρ such that $\lambda_M(H) \le -\rho$ and condition (19) is satisfied, and moreover,*

$$s_i < \rho(2+\eta_i^{-1}), (1 \le i \le n), \tag{23}$$

where H is defined by (18), then the conclusions of Theorem 2 hold.

Corollary 4. *Under conditions (11) and (22), if*

$$2b_i > \sum_{j=1}^{n} |d_{ij}| + \sum_{j=1}^{n} |\tilde{a}_{ij}| + \sum_{j=1}^{n} |d_{ji}| + \sum_{j=1}^{n} |\tilde{a}_{ji}|\eta_i^{-1} + s_i, \tag{24}$$

then the conclusions of Theorem 2 hold.

Remark 2. In [1,2,5], it is all assumed that the discrete delays are bounded, here, by introducing the decay factor $e^{-\varepsilon\delta_i(t)}$, our results can be used in the case of unbounded delays.

4 A Two-Dimensional Example

Consider the following two-neuron stochastic neural networks:

$$dx(t) = \Big[-Bx(t)+AG(t,y(t))+\int_{-\infty}^{0} \frac{Dx(t+\theta)}{(1-\theta)^2}d\theta\Big]dt+EG(t,y(t))dw(t), \tag{25}$$

where $y(t) = x(t-\delta(t))$, $\delta(t) \in C^1(\mathbb{R}_+;\mathbb{R}_+)$ is variable delay, and

$$B = \begin{bmatrix} 6 & 0 \\ 0 & 8 \end{bmatrix}, A = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}, D = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, E = \begin{bmatrix} a & 0 \\ 0 & 0.2 \end{bmatrix},$$

where a is a positive constant. Let $d\mu(\theta) = (1-\theta)^{-2}d\theta$, then $\mu(\theta) \in M_\varepsilon$ $(0 < \varepsilon < 1)$. For simplicity, let $\eta_i = 1/2$, $\beta_i = 1 (i = 1, 2)$. From (13) we have $s_1 = 2a^2$, $s_2 = 0.08$.

(i) *Application of Corollary 3*. Choose $\bar{q}_1 = \tilde{q}_1 = 3$, $\bar{q}_2 = \tilde{q}_2 = 4$. By (18) we can compute

$$H = \begin{pmatrix} -3 & 0 & 0 & 1 & 1 & 1 \\ 0 & -4 & 0 & 0 & 1 & 1 \\ 0 & 0 & -3 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 0 & 0 \\ 1 & 1 & 0 & 0 & -3 & 0 \\ 1 & 1 & 0 & 0 & 0 & -4 \end{pmatrix}.$$

This can derive $\lambda_M(H) = -1.2375$, which implies $\rho = 1.2375$. Substituting the parameters into (23) gives $2a^2 \vee 0.08 < 4\rho$, which shows $0 < a < 1.5732$. Applying Corollary 3 we have, when $0 < a < 1.5732$, there exists $q > 0$, for any given initial data $\xi \in C(q, 2)$, Eq.(25) admits a unique global solution $x(t, \xi)$ which satisfies (9) and (10).

(ii) *Application of Corollary 4*. Substituting the parameters into (24) gives $0 < a < 1.5811$. Applying Corollary 4 we have, when $0 < a < 1.5811$, there exists $q > 0$, for any given initial data $\xi \in C(q, 2)$, Eq.(25) admits a unique global solution $x(t, \xi)$ which satisfies (9) and (10). Obviously, Corollary 4 is more convenient in application.

Choose the initial data $\xi(s) = 1$, $s \in (-\infty, 0]$, $\varepsilon = 0.5$ and $G(t, x) = x(0.5t)e^{-0.5\times0.5t}$, Fig.1 shows the trivial solution of Eq. (25) is exponentially stable in mean square.

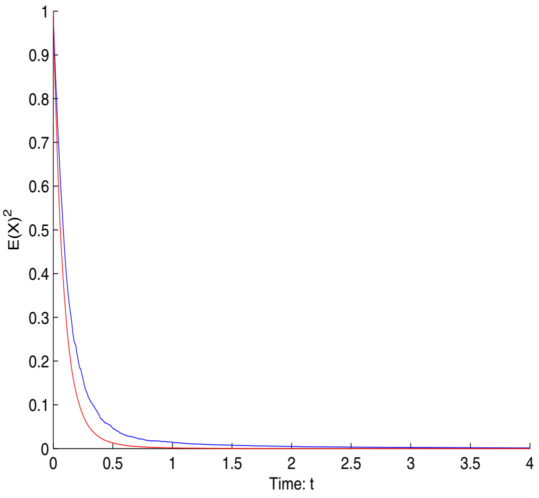


Fig. 1. Numerical simulation of (25)

Acknowledgments. The authors would like to thank the referees for their helpful suggestions and comments.

References

1. Balasubramaniam, P., Rakkiyappan, R.: Global Asymptotic Stability of Stochastic Recurrent Neural Networks with Multiple Discrete Delays and Unbounded Distributed Delays. Applied Mathematics and Computation 204, 680–686 (2008)
2. 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)
3. Cao, J., Zhou, D.: Stability Analysis of Delayed Cellular Neural Networks. Neural Networks 11, 1601–1605 (1998)

4. Huang, C., He, Y., Wang, H.: Mean Square Exponential Stability of Stochastic Recurrent Neural Networks with Time-varying Delays. Computers and Mathematics with Applications 56, 1773–1778 (2008)
5. Li, X., Fu, X.: Stability Analysis of Stochastic Functional Differential Equations with Infinite Delay and its Application to Recurrent Neural Networks. Journal of Computational and Applied Mathematics 234(2), 407–417 (2010)
6. Liao, X., Mao, X.: Exponential Stability and Instability of Stochastic Neural Networks. Stochastic. Anal. Appl. 14(2), 165–185 (1996)
7. Liao, X., Mao, X.: Stability of Stochastic Neural Networks. Neural, Parallel Sci. Comput. 4(2), 205–224 (1996)
8. Liu, Y., Wang, Z., Liu, X.: Stability Criteria for Periodic Neural Networks with Discrete and Distributed Delays. Nonlinear Dyn. 4, 93–103 (2007)
9. 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)
10. Mao, X.: Stochastic Differential Equations and Applications. Horwood, Chichester (1997)
11. Mohamad, S., Gopalsamy, K.: Exponential Stability of Continuous-time and Discrete-time Cellular Neural Networks with Delays. Appl. Math. Comput. 135, 17–38 (2003)
12. Roska, T., Wu, C.W., Balsi, M., Chua, L.O.: Stability and Dynamics of Delay-type General Cellular Neural Networks. IEEE Trans. Circuit Syst. I 39(6), 487–490 (1992)
13. Sun, Y., Cao, J.: pth Moment Exponential Stability of Stochastic Recurrent Neural Networks with Time-varying Delays. Nonlinear Analysis: Real Word Applications 8, 1171–1185 (2007)
14. Tank, D.W., Hopfield, J.J.: Neural Computation by Concentrating Information in Time. Proc. Acad. Sci. USA 84, 1896–1900 (1987)
15. Wang, Z., Liu, Y., Liu, X.: On Global Asymptotic Stability of Neural Netwoks with Discrete and Distributed Delays. Phys. Lett. A 345, 299–308 (2005)
16. Zeng, Z., Wang, J., Liao, X.: Global Asymptotic Stability and Global Exponential Stability of Neural Networks with Unbounded Time-varying Delays. IEEE Trans. Circuits Syst. II, Express Briefs 52(3), 168–173 (2005)

Stability of Neural Networks with Both Impulses and Time-Varying Delays on Time Scale

Yupei Lv[1], Bo Zhou[2], and Qiankun Song[2]

[1] Department of Mathematics, Huzhou Teachers College, Huzhou 313000, China
[2] Department of Mathematics, Chongqing Jiaotong University, Chongqing 400074, China
qiankunsong@163.com

Abstract. In this paper, the stability of neural networks with both impulses and time-varying delays on time scale is investigated, the existence of Delta derivative of time-varying delays is not assumed. By employing time scale calculous theory, free weighting matrix method and linear matrix inequality (LMI) technique, a delay-dependent sufficient condition is obtained to ensure the stability of equilibrium point for neural networks with both impulses and time-varying delays on time scale. An example with simulations is given to show the effectiveness of the theory.

Keywords: Stability, Neural networks, Time-varying delays, Impulses, Time scale.

1 Introduction

The theory of time scale, which was first introduced and studied by S.Hilger [1], has a tremendous potential for applications in some mathematical models of real processes and phenomena studied in automatic control [2], population dynamics [3], economics [4] and so on. This novel and fascinating type of mathematics is more general and versatile than the traditional theory of differential and difference equations as it can mathematically describe continuous and discrete dynamical equations under the same framework [5]. As it is well known, both continuous and discrete systems are very important in applications, but it is troublesome to study the stability for continuous and discrete systems, respectively. Therefore, it is significant to study the systems on time scale where the continuous and discrete situations are unified [6].

Besides delay effect, impulsive effect is also likely to exist in neural networks. For instance, in implementation of electronic networks, the state of the networks is subject to instantaneous perturbations and experiences abrupt change at certain instants, which may be caused by switching phenomenon, frequency change or other sudden noise, that is, it exhibits impulsive effects [7]. Therefore, it is necessary to consider both impulsive effect and delay effect on dynamical behaviors of neural networks. Some results of impulsive effect on dynamics of delayed neural networks have been gained, see [7-9] and the references therein.

Recently, some delayed neural networks on time scale were considered, the dynamics of neural networks with constant delays on time scale were analyzed [10-15]. In

© Springer-Verlag Berlin Heidelberg 2011

[10], authors investigated the stability of equilibrium point for neural networks with constant delays on time scale, and give a sufficient condition for checking the stability of equilibrium point. In [11-13], the problem on existence and exponential stability of periodic solution was considered, several criteria for checking the existence and exponential stability of periodic solution were obtained. In [14,15], the impulsive effect on dynamics of delayed neural networks on time scale were analyzed. It is worth to pointing out that, the stability results in [15] are delay-independent. It means that any information on the size of delays is included in criteria. It is known that delay-dependent stability conditions, which employ the information on the size of delays, are generally less conservative than delay-independent ones especially when the size of the delay is small [16]. So, it is important to give delay-dependent stability conditions for delayed neural networks on time scale

Motivated by the above discussions, the object of this paper is to study the stability for neural networks with both time-varying delays and impulses on time scale. The time-varying delays are assumed to be bounded but not necessarily differentiable. The proposed stability condition in this paper is expressed in terms of LMI, which is easy to be checked by recently developed algorithms solving LMIs. Furthermore, A simulation example is given to show the effectiveness and less conservatism of the obtained conditions.

Notations: The notations are quite standard. The subscript T denotes the matrix transposition. The notation $X \geq Y$ (respectively, $X > Y$) means that $X - Y$ is positive semi-definite (respectively, positive definite). $\rho_{\max}(A)$ and $\rho_{\min}(A)$ is defined as the maximum and minimum eigenvalue of matrix A, respectively. Set $[-\tau, b]_{\mathbb{T}}$ is defined as: $[-\tau, b]_{\mathbb{T}} := \{t \in \mathbb{T}, -\tau \leq t \leq b\}$. And we also have that, the time scale $\mathbb{T}$ has a bounded graininess $\mu(t) \leq \overline{\mu} < \infty$. Sometimes, the arguments of a function or a matrix will be omitted in the analysis when no confusion can arise.

2 Problem Formulation and Preliminaries

In this paper, we consider the following neural network model on time scale $\mathbb{T}$. $\mathbb{T}$ is an arbitrary nonempty closed subset of $\mathbb{R}$.

$$\begin{cases} x^{\Delta}(t) = -Cx(t) + Df^{*}(x(t)) + Ef^{*}(x(t-\tau(t))) + J & t > 0, t \neq t_k, \\ \Delta x(t_k) = I(x(t_k)) & k \in \mathbb{Z}^{+}, \end{cases} \tag{1}$$

for $t \in \mathbb{T}$, where $x^{\Delta}(t)$ is the Delta derivative of function $x(t)$, which means that

$$x^{\Delta}(t) = \begin{cases} \lim_{s\to t} \frac{x(t)-x(s)}{t-s}, & \sigma(t) = t, \\ \frac{x(\sigma(t))-x(t)}{\sigma(t)-t}, & \sigma(t) > t, \end{cases}$$

with $\sigma(t) := \inf\{s \in \mathbb{T} : s > t\}$.

The initial conditions of system (1) are of the forms

$$x_i(s) = \phi_i(s), \quad s \in [-\tau, 0]_{\mathbb{T}},$$

with $\phi_i(s)$ is bounded and continuous on $[-\tau, 0]_{\mathbb{T}}$.

In system (1), $x(t) = (x_1(t), x_2(t), \cdots, x_n(t))^T$, $x_i(t)$ is the state of ith neuron at time t. $f^*(x(t-\tau(t))) = (f_1^*(x_1(t-\tau_1(t))), f_2^*(x_2(t-\tau_2(t))), \cdots, f_n^*(x_n(t-\tau_n(t))))^T$, $f_i^*(x_i(t_k))$ is the activation function of ith neuron. $I(x(t_k)) = (I_1(x_1(t_k)), I_2(x_2(t_k))^T, \cdots, I_n(x_n(t_k)))$, $I_i(\cdot)$ is the impulsive operator with $I_i(x_i(t_k)) = x_i(t_k^+) - x_i(t_k^-)$. $J = (J_1, J_2, \cdots, J_n)^T$ is the constant input vector. $\tau(t)$ responds to the discrete delay of neural networks which satisfies if $t \in \mathbb{T}$, then $t - \tau(t) \in \mathbb{T}$ and $0 \le \tau(t) \le \tau$ (τ is a constant). $C = \mathrm{diag}(c_1, c_2, \cdots, c_n)$, $d = (d_{ij})_{n\times n}$ and $E = (e_{ij})_{n\times n}$ stand for the interconnection matrices representing the weight coefficients of the neurons. $\varDelta x(t_k) = I(x(t_k)) = x(t_k^+) - x(t_k^-)$ is the impulse at the moment t_k, t_k is a strictly increasing sequence such that $\lim_{k\to+\infty} t_k = +\infty$.

In the following, we will give several useful definitions.

Definition 1. *([10]) Let $\mathbb{T}$ be a time scale, we define the graininess μ: $\mathbb{T} \to \mathbb{R}^+$, by*

$$\mu(t) = \sigma(t) - t.$$

Definition 2. *System (1) is globally stable on time scale $\mathbb{T}$, if there exits a positive constant $M(\tau)$ such that the solution $x(t) = (x_1(t), \cdots, x_n(t))$ of system (1) satisfies*

$$\| x(t) \| \le M(\tau) \max\{\| \phi(s) \|_{\mathbb{T}}, \| \phi^{\varDelta}(s) \|_{\mathbb{T}}\},$$

with $\| \phi(s) \|_{\mathbb{T}} = \sup_{s\in[-\tau,0]_{\mathbb{T}}} \| \phi(s) \|$ *and* $\| \phi^{\varDelta}(s) \|_{\mathbb{T}} = \sup_{s\in[-\tau,0]_{\mathbb{T}}} \| \phi^{\varDelta}(s) \|$.

Definition 3. *([10]) Function f is continuous, if $F^{\varDelta}(t) = f(t)$, we define the Delta integral by*

$$\int_a^t f(s)\varDelta s = F(t) - F(a),$$

To prove our results, the following lemmas are necessary.

Lemma 1. *([9]) If f and g are two differentiable functions on time scale $\mathbb{T}$, then the following formula holds*

$$(fg)^{\varDelta}(t) = f^{\varDelta}(t)g(t) + f(\sigma(t))g^{\varDelta}(t) = g^{\varDelta}(t)f(t) + g(\sigma(t))f^{\varDelta}(t).$$

Lemma 2. *(Schur complement). Given constant matrices P, Q and R, there $P^T = P$, $Q^T = Q$, then*

$$\begin{bmatrix} P & R \\ R^T & -Q \end{bmatrix} < 0,$$

is equivalent to the following conditions $Q > 0$ and $P + RQ^{-1}R^T < 0$.

Throughout this paper, we make the following assumptions.

(H1) For any $i \in \{1, 2, \cdots, n\}$, the activation function $f_i(\cdot)$ is bounded on $\mathbb{R}$.

(H2) For any $i \in \{1, 2, \cdots, n\}$ and $\alpha_1 \neq \alpha_2$, the activation function $f_i(\cdot)$ satisfies

$$F_i^- \le \frac{f_i^*(\alpha_1) - f_i^*(\alpha_2)}{\alpha_1 - \alpha_2} \le F_i^+,$$

where F_i^- and F_i^+ are some constants.

(H3) At the points of discontinuity t_k, $x(t_k) = x(t_k - 0)$ and $x^{\Delta}(t_k) = x^{\Delta}(t_k - 0)$.

(H4) If x^* is the equilibrium point of system (1), then the impulsive operator satisfies $I(x(t_k)) = -\gamma_k(x(t_k) - x^*)$, $0 < \gamma_k < 2$.

3 Main Results

For presentation convenience, we denote

$$F_1 = \mathrm{diag}(F_1^- F_1^+, \cdots, F_n^- F_n^+), \quad F_2 = \mathrm{diag}(\frac{F_1^- + F_1^+}{2}, \cdots, \frac{F_n^- + F_n^+}{2}).$$

Theorem 1. *If Assumption (H1)-(H4) hold, system (1) is globally stable at equilibrium point on time scale $\mathbb{T}$, if there exist four symmetric positive definite matrices P, Q, Q_1, Q_2, two positive diagonal matrices $R = \mathrm{diag}(r_1, r_2, \ldots, r_n)$ and $H = \mathrm{diag}(h_1, h_2, \ldots, h_n)$, twelve any appropriately dimensioned matrices N_i, M_i $(i = 1, 2, \ldots, 6)$, such that the following LMI holds*

$$\begin{bmatrix} \Omega + \Pi + \Pi^T & \sqrt{\tau}N & \sqrt{\tau}M \\ * & -Q_2 & 0 \\ * & * & -Q_2 \end{bmatrix} < 0, \tag{2}$$

where

$N = [N_1^T, N_2^T, N_3^T, N_4^T, N_5^T, N_6^T]^T$, $M = [M_1^T, M_2^T, M_3^T, M_4^T, M_5^T, M_6^T]^T$,
$\Pi = [N, 0, 0, M - N, 0, -M]$,

$$\Omega = \begin{bmatrix} \Omega_{11} & \Omega_{12} & \Omega_{13} & 0 & 0 & 0 \\ * & \Omega_{22} & \Omega_{23} & 0 & 0 & 0 \\ * & * & \Omega_{33} & 0 & 0 & 0 \\ * & * & * & \Omega_{44} & \Omega_{45} & 0 \\ * & * & * & * & \Omega_{55} & 0 \\ * & * & * & * & * & \Omega_{66} \end{bmatrix}$$

with

$\Omega_{11} = \bar{\mu}C^T PC + \tau C^T Q_2 C + Q_1 - PC - C^T P - F_1 R,$
$\Omega_{12} = -\bar{\mu}C^T PD - \tau C^T Q_2 D + \frac{1}{2}(PD + D^T P) + F_2 R,$
$\Omega_{13} = -\bar{\mu}C^T PE - \tau C^T Q_2 E + \frac{1}{2}(PE + E^T P),$
$\Omega_{22} = \bar{\mu}D^T PD + Q + \tau D^T Q_2 D - R$, $\Omega_{23} = \bar{\mu}D^T PE + \tau D^T Q_2 E,$
$\Omega_{33} = \bar{\mu}E^T PE + \tau E^T Q_2 E - H$, $\Omega_{34} = F_2 H$, $\Omega_{44} = -F_1 H$, $\Omega_{55} = -Q$, $\Omega_{66} = -Q_1$.

Proof. Due to the boundedness of activation function, system (1) has a unique equilibrium point $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$ [7].

First, we consider $t > 0$, $t \neq t_k$ for all $k = 1, 2, \cdots$.

Let $w(t) = x_i(t) - x_i^*$ and $f_i(w_i(t)) = f_i^*(w_i(t) + x_i^*) - f_i^*(x_i^*)(i = 1, 2, \ldots, n)$, system (1) without impulses can be rewritten into

$$w^{\Delta}(t) = -Cw(t) + Df(w(t)) + Ef(w(t - \tau(t))), \tag{3}$$

where $f(w(t)) = [f_1(w_1(t)), f_2(w_2(t)), \ldots, f_n(w_n(t))]^T$.

By Assumption (H2), it is easy to see that

$$F_j^- \leq \frac{f_j(w_j(t))}{w_j(t)} \leq F_j^+,$$

for all $w_j(t) \neq 0$.

From Assumption (H2), the following inequality holds,

$$\begin{bmatrix} w(t) \\ f(w(t)) \end{bmatrix}^T \begin{bmatrix} F_i^- F_i^+ e_i e_i^T & -\frac{F_i^- + F_i^+}{2} e_i e_i^T \\ -\frac{F_i^- + F_i^+}{2} e_i e_i^T & e_i e_i^T \end{bmatrix} \begin{bmatrix} w(t) \\ f(w(t)) \end{bmatrix} \leq 0, \quad i = 1, 2, \cdots, n,$$

where e_r denotes the unit column vector having 1 element on its rth row and zeros elsewhere.

Let $R = \mathrm{diag}(r_1, r_2, \cdots, r_n)$,, $H = \mathrm{diag}(h_1, h_2, \cdots, h_n)$, we have

$$\alpha_1(t) = \begin{bmatrix} w(t) \\ f(w(t)) \end{bmatrix}^T \begin{bmatrix} F_1 R & -F_2 R \\ -F_2 R & R \end{bmatrix} \begin{bmatrix} w(t) \\ f(w(t)) \end{bmatrix} \leq 0.$$

Similarly, we have

$$\alpha_2(t) = \begin{bmatrix} w(t-\tau(t)) \\ f(w(t-\tau(t))) \end{bmatrix}^T \begin{bmatrix} F_1 H & -F_2 H \\ -F_2 H & H \end{bmatrix} \begin{bmatrix} w(t-\tau(t)) \\ f(w(t-\tau(t))) \end{bmatrix} \leq 0.$$

Let $N = [N_1^T, N_2^T, N_3^T, N_4^T, N_5^T, N_6^T]^T$ and $M = [M_1^T, M_2^T, M_3^T, M_4^T, M_5^T, M_6^T]^T$. From Newton-Leibniz formula, we obtain

$$\alpha_3(t) = 2\gamma^T(t) M [w(t-\tau(t)) - w(t-\tau) - \int_{t-\tau}^{t-\tau(t)} w^{\Delta}(s) \Delta s],$$

$$\alpha_4(t) = 2\gamma^T(t) N [w(t) - w(t-\tau(t)) - \int_{t-\tau(t)}^{t} w^{\Delta}(s) \Delta s],$$

where $\gamma(t) = [w^T(t), f^T(w(t)), f^T(w(t-\tau(t)), w(t-\tau(t)), f^T(w(t-\tau)), w(t-\tau)]$.

Consider the following Lyapunov-Krasovskii functional candidate

$$V(t) = w^T(t) P w(t) + \int_{t-\tau}^{t} f^T(w(s)) Q f(w(s)) \Delta s + \int_{t-\tau}^{t} w^T(s) Q_1 w(s) \Delta s \qquad (4)$$
$$+ \int_{t-\tau}^{t} \int_{\theta}^{t} (w^{\Delta}(s))^T Q_2 w^{\Delta}(s) \Delta s \Delta \theta.$$

Calculate the Delta derivative of $V(t)$ along the trajectories of system (1), we obtain

$$\begin{aligned}
V^{\Delta}(t) &\leq w^T(t)(\overline{\mu}C^TPC + \tau C^TQ_2C + Q_1 - PC - C^TP)w(t)\\
&\quad + w^T(t)(-\overline{\mu}C^TPD - \tau C^TQ_2D + \frac{1}{2}(PD + D^TP))f(w(t))\\
&\quad + w^T(t)(-\overline{\mu}C^TPE - \tau C^TQ_2E + \frac{1}{2}(PE + E^TP))f(w(t-\tau(t)))\\
&\quad + f^T(w(t))(\overline{\mu}D^TPD + Q + \tau D^TQ_2D)f(w(t))\\
&\quad + f^T(w(t))(\overline{\mu}D^TPE + \tau D^TQ_2E)f(w(t-\tau(t)))\\
&\quad + f^T(w(t-\tau(t)))(\overline{\mu}E^TPE + \tau E^TQ_2E)f(w(t-\tau(t)))\\
&\quad - f^T(t-\tau)Qf(w(t-\tau)) - w^T(t-\tau)Q_1w(t-\tau) - \alpha(1) - \alpha(2) + \alpha(3) + \alpha(4).\\
&\leq \gamma^T(t)[\Omega + \Pi + \Pi^T + \tau NQ_2^{-1}N^T + \tau MQ_2^{-1}M^T]\gamma(t)\\
&\quad -[\int_{t-\tau(t)}^{t}(N^T\gamma(t) + Q_2w^{\Delta}(s))^TQ_2^{-1}(N^T\gamma(t) + Q_2w^{\Delta}(s))\Delta s]\\
&\quad -[\int_{t-\tau}^{t-\tau(t)}(M^T\gamma(t) + Q_2w^{\Delta}(s))^TQ_2^{-1}(M^T\gamma(t) + Q_2w^{\Delta}(s))\Delta s]\\
&\leq \gamma^T(t)[\Omega + \Pi + \Pi^T + \tau NQ_2^{-1}N^T + \tau MQ_2^{-1}M^T]\gamma(t).
\end{aligned}$$

Let $\Theta = \Omega + \Pi + \Pi^T + \tau NQ_2^{-1}N^T + \tau MQ_2^{-1}M^T$. If $\Theta < 0$, by Lemma 2, $\Theta < 0$ is equivalent to the following linear matrix inequality

$$\begin{bmatrix} \Omega + \Pi + \Pi^T & \sqrt{\tau}N & \sqrt{\tau}M \\ * & -Q_2 & 0 \\ * & * & -Q_2 \end{bmatrix} < 0,$$

It is easy to see that $\Theta < 0$ is implied by LMI condition (2), in other words, LMI condition (2) implies the following inequality holds

$$V^{\Delta}(t) \leq 0, \quad t \in \mathbb{T}. \tag{5}$$

Then, from Definition 3, formula (5) is equivalent to

$$V(t) \leq V(0). \tag{6}$$

On the other hand, if $t = t_k$ for all $k = 1, 2, \cdots$, we can deduce that

$$\begin{aligned}
V(t_k + 0) - V(t_k) &= w^T(t_k + 0)Pw(t_k + 0) + \int_{t_k+0-\tau}^{t_k+0} f^T(w(s))Qf(w(s))\Delta s\\
&\quad + \int_{t_k+0-\tau}^{t_k+0} w^T(s)Q_1w(s)\Delta s + \int_{t_k+0-\tau}^{t_k+0}\int_{\theta}^{t_k+0}(w^{\Delta}(s))^TQ_2w^{\Delta}(s)\Delta s\Delta\theta\\
&\quad -(w^T(t_k)Pw(t_k) + \int_{t_k-\tau}^{t_k} f^T(w(s))Qf(w(s))\Delta s + \int_{t_k-\tau}^{t_k} w^T(s)Q_1w(s)\Delta s\\
&\quad + \int_{t-\tau}^{t_k}\int_{\theta}^{t_k}(w^{\Delta}(s))^TQ_2w^{\Delta}(s)\Delta s\Delta\theta)\\
&= \gamma_k(\gamma_k - 2)w^T(t_k)Pw(t_k).
\end{aligned} \tag{7}$$

Since $0 < \gamma_k < 2$, we know that $V(t_k + 0) \leq V(t_k)$. It follows that $V(t) < V(0)$ for $t > 0$.
Now, we denote

$$W = \mathrm{diag}(L_1, L_2, \cdots, L_n),$$

with $L_i = \max\{|F_i^+|, |F_i^-|\}$.
From (4) and (6), we obtain

$$\begin{aligned} V(0) &\leq w^T(0)Pw(0) + \int_{-\tau}^{0} f^T(w(s))Qf(w(s))\Delta s + \int_{-\tau}^{0} w^T(s)Q_1 w(s)\Delta s \\ &\quad + \int_{-\tau}^{0}\int_{-\tau}^{0} (w^{\Delta}(s))^T Q_2 w^{\Delta}(s)\Delta s\Delta\theta \\ &\leq \rho_{max}(P) \parallel \phi(s) \parallel_{\mathbb{T}} + \rho_{max}(W^TQW) \parallel \phi(s) \parallel_{\mathbb{T}} + \tau\rho_{max}(Q_1) \parallel \phi(s) \parallel_{\mathbb{T}} \\ &\quad + \tau^2\rho_{max}(Q_2) \parallel \phi^{\Delta}(s) \parallel_{\mathbb{T}} . \end{aligned} \tag{8}$$

Let $N_1(\tau) = \max\{\rho_{max}(P), \rho_{max}(W^TQW), \tau\rho_{max}(Q_2)\}$, $N_2(\tau) = \tau^2\rho_{max}(Q_2)$, we can get

$$V(0) \leq N_1(\varepsilon) \parallel \phi(s) \parallel_{\mathbb{T}} + N_2(\varepsilon) \parallel \phi^{\Delta}(s) \parallel_{\mathbb{T}} .$$

It is obvious that $V(t) \geq \rho_{min}(P) \parallel w(s) \parallel$, we obtain

$$\parallel w(s) \parallel \leq \frac{N_1(\tau)}{\rho_{min}(P)} \parallel \phi(s) \parallel_{\mathbb{T}} + \frac{N_2(\tau)}{\rho_{min}(P)} \parallel \phi^{\Delta}(s) \parallel_{\mathbb{T}} .$$

Let $M(\varepsilon) = \max\{\frac{N_1(\tau)}{\rho_{min}(P)}, \frac{N_2(\tau)}{\rho_{min}(P)}\}$. We can finally get

$$\parallel w(t) \parallel \leq M(\tau) \max\{\parallel \phi(s) \parallel_{\mathbb{T}}, \parallel \phi^{\Delta}(s) \parallel_{\mathbb{T}}\}.$$

Therefore, from Definition 2, system (3) is globally stable at origin on time scale $\mathbb{T}$, which guarantees the global stability of system (1) at equilibrium point on time scale $\mathbb{T}$. The proof is completed.

4 Example

The following example is given to show the effectiveness of the delay-dependent global stability result in Theorem 1.

Example 1. Consider a time-varying delayed neural network model with impulses on a special time scale $\mathbb{T} = \bigcup_{k=0}^{+\infty}[2k, 2k+1]$ with the following parameters

$$C = \begin{bmatrix} 0.7 & 0 \\ 0 & 0.8 \end{bmatrix}, \quad D = \begin{bmatrix} -0.2 & -0.2 \\ -0.1 & -0.1 \end{bmatrix}, \quad E = \begin{bmatrix} -0.15 & 0.2 \\ -0.1 & 0.15 \end{bmatrix},$$

$$J = [0.88, 0.64]^T, f_1(x) = \frac{1}{20}(|x+1| - |x-1|), \quad f_2(x) = \frac{1}{10}(|x+1| - |x-1|),$$
$$\tau(t) = 0.1|\sin(t)|.$$

The impulse operator is chosen as $I(x_1(t_k)) = -(1 + \sin(t_k))(x_1(t_k) - 1.1972)$ and $I(x_2(t_k)) = -(1 + \cos(t_k))(x_2(t_k) - 0.7826)$, respectively.

It is easy to check that the graininess $\mu(t)$ of $\mathbb{T}$ satisfies $0 \leq \mu(t) \leq 1$; the time-varying delays $\tau(t)$ are continuous with upper bound $\tau = 0.1$. However $\tau(t)$ is not differentiable at $t = k\pi (k = 0, \pm 1, \pm 2, \ldots)$.

It can be verified that Assumption (**H1**) is satisfied with $F_1^- = -0.1$, $F_1^+ = 0.1$, $F_2^- = -0.2$, $F_2^+ = 0.2$. Thus

$$F_1 = \begin{bmatrix} -0.01 & 0 \\ 0 & -0.04 \end{bmatrix}, \quad F_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

By employing Matlab LMI Toolbox, we can get the solution of LMI condition (2) as follows, which guarantees the stability for system (1) at equilibrium point.

$$P = \begin{bmatrix} 28.3335 & -15.0795 \\ -15.0795 & 50.9139 \end{bmatrix}, \quad Q = \begin{bmatrix} 7.1186 & -0.6671 \\ -0.6671 & 6.8731 \end{bmatrix},$$

$$Q_1 = \begin{bmatrix} 18.3513 & -7.0831 \\ -7.0831 & 29.5969 \end{bmatrix}, \quad Q_2 = \begin{bmatrix} 26.5456 & -0.1513 \\ -0.1513 & 26.6786 \end{bmatrix},$$

$$M_1 = \begin{bmatrix} -2.8279 & 0.1344 \\ 0.1344 & -3.1357 \end{bmatrix}, \quad M_2 = \begin{bmatrix} -5.6673 & 0.3653 \\ -0.0483 & -5.7382 \end{bmatrix}, \quad M_3 = \begin{bmatrix} 0.1598 & -0.8102 \\ 0.7231 & -0.1779 \end{bmatrix},$$

$$M_4 = \begin{bmatrix} -0.6916 & 2.8220 \\ 2.8220 & -5.3000 \end{bmatrix}, \quad M_5 = \begin{bmatrix} -2.3969 & -0.0814 \\ -0.0814 & -2.4268 \end{bmatrix}, \quad M_6 = \begin{bmatrix} 3.5850 & 3.2385 \\ 3.2385 & -1.5651 \end{bmatrix}$$

$$N_1 = \begin{bmatrix} -9.7959 & -3.2385 \\ -3.2385 & -4.8423 \end{bmatrix}, N_2 = \begin{bmatrix} -11.9014 & 0.7672 \\ -0.1014 & -12.0503 \end{bmatrix}. N_3 = \begin{bmatrix} 0.3356 & -1.7015 \\ 1.5185 & -0.3735 \end{bmatrix},$$

$$N_4 = \begin{bmatrix} 12.3514 & 2.8220 \\ 2.8220 & 8.1556 \end{bmatrix}, N_5 = \begin{bmatrix} 2.3969 & 0.0814 \\ 0.0814 & 2.4268 \end{bmatrix}, \quad N_6 = \begin{bmatrix} 3.3831 & 0.1344 \\ 0.1344 & 3.2717 \end{bmatrix}$$

$$R = \begin{bmatrix} 20.2026 & 0 \\ 0 & 19.7715 \end{bmatrix}, H = \begin{bmatrix} 27.5439 & 0 \\ 0 & 28.2730 \end{bmatrix},$$

The time responses of system (1) with or without impulses on $\mathbb{T}$ are given, please see Fig 1 and Fig 2, respectively. The initial states of system (1) are $\phi_1(s) = \sin(s)$ and $\phi_2(s) = \cos(s)$, $s \in [-0.1, 0]_{\mathbb{T}}$.

5 Conclusions

This paper has proposed an LMI formed condition to ensure the stability of neural networks with impulses on time scale by employing time scale calculus, linear matrix technique and free weight matrix method. The differentiability on the time-varying delays were not required. The developed stability conditions are in terms of LMIs, which can be checked easily by recently developed algorithms solving LMIs. Simulation example is employed to illustrate the theories.

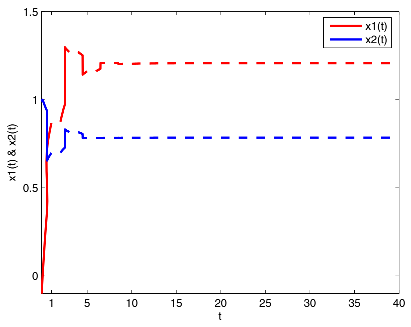


Fig. 1. Time responses with impulses

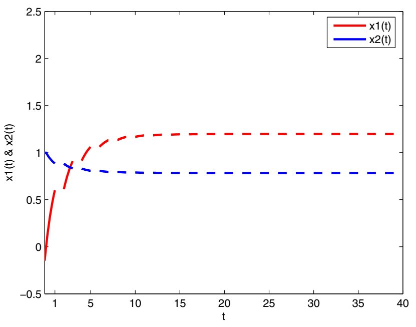


Fig. 2. Time responses without impulses

Acknowledgments

The work is supported by the National Natural Science Foundation of China under Grant 60974132, the Foundation of the Educational Committee of Zhejiang Province under Grants Y200805602 and Y201017330, and the Postgraduate Education Innovation Foundation of Chongqing Jiaotong University.

References

1. Hilger, S.: Analysis on Measure Chains-A Unified Approach to Continuous and Discrete Calculus. Results in Mathematics 18, 18–56 (1990)
2. Hilscher, R., Zeidan, V.: Weak Maximum Principle and Accessory Problem for Control Problems on Time Scales. Nonlinear Analysis 70, 3209–3226 (2009)
3. Du, B., Hu, X.P., Ge, W.G.: Periodic Solution of A Neutral Delay Model of Single-Species Population Growth on Time Scales. Communications in Nonlinear Science and Numerical Simulation 15, 394–400 (2010)
4. Aticia, F.M., Bilesa, D.C., Lebedinskyb, A.: An Application of Time Scales to Economics. Mathematical and Computer Modelling 43, 718–726 (2006)
5. Agarwal, R., Bohner, M., O'Regan, D., Peterson, A.: Dynamic Equations on Time Scales: A Survey. Journal of Computation and Applied Mathematics 141, 1–26 (2002)
6. Bohner, M., Peterson, A.: Dynamic Equations on Time Scales an Introduction with Applications. Birkhäuser, Basel (2001)
7. Xu, D.Y., Yang, Z.C.: Impulsive Delay Differential Inequality and Stability of Neural Networks. Journal of Mathematical Analysis and Applications 305, 107–120 (2005)
8. Song, Q.K., Cao, J.D.: Impulsive Effects on Stability of Fuzzy Cohen-Grossberg Neural Networks With Time-Varying Delays. IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics 37, 733–741 (2007)
9. Zhang, H.G., Dong, M., Wang, Y.C., Sun, N.: Stochastic Stability Analysis of Neutral-Type Impulsive Neural Networks with Mixed Time-Varying Delays and Markovian Jumping. Neurocomputing 73, 2689–2695 (2010)
10. Chen, A.P., Du, D.J.: Global Exponential Stability of Delayed BAM Network on Time Scale. Neurocomputing 71, 3582–3588 (2008)
11. Li, Y.K., Zhao, L.L., Liu, P.: Existence and Exponential Stability of Periodic Solution of High-Order Hopfield Neural Network with Delays on Time Scales. Discrete Dynamics in Nature and Society 2009, Article ID 573534 (2009)

12. Chen, A.P., Chen, F.L.: Periodic Solution to BAM Neural Network with Delays on Time Scales. Neurocomputing 73, 274–282 (2009)
13. Zheng, F.Y., Zhou, Z., Ma, C.: Periodic Solutions for A Delayed Neural Network Model on A Special Time Scale. Applied Mathematics Letters 23, 571–575 (2010)
14. Li, Y.K., Chen, X.R., Zhao, L.: Stability and Existence of Periodic Solutions to Delayed Cohen-Grossberg BAM Neural Networks with Impulses on Time Scales. Neurocomputing 72, 1621–1630 (2009)
15. Li, Y.K., Hua, Y.H., Fei, Y.: Global Exponential Stability of Delayed Cohen-Grossberg BAM Neural Networks with Impulses on Time Scales. Journal of Inequalities and Applications 2009, 1–17 (2009)
16. Xu, S.Y., Lam, J., Ho, D.W.C., Zou, Y.: Novel Global Asymptotic Stability Criteria for Delayed Cellular Neural Networks. IEEE Transactions on Circuits and Systems-II: Express Briefs 52, 349–353 (2005)

Synchronization of Nonidentical Chaotic Neural Networks with Time-Varying Delays

Qiankun Song

Department of Mathematics, Chongqing Jiaotong University,
Chongqing 400074, China
qiankunsong@163.com

Abstract. In this paper, the problem on synchronization of two nonidentical chaotic neural networks with time-varying delays is investigated. By constructing a proper sliding surface, and employing a combination of the free-weighting matrix method, inequality technique and Lyapunov theory, a sliding mode controller is designed to achieve the asymptotical synchronization of two nonidentical chaotic neural networks with time-varying delays. The provided condition is expressed in terms of linear matrix inequalities (LMIs), and is dependent on the time delay. A simulation example is given to show the effectiveness.

Keywords: Synchronization, Chaotic neural networks, Time-varying delays, Integral sliding mode control.

1 Introduction

Since the drive-response concept for considering synchronization of two chaotic systems was proposed in 1990 [1], the synchronization of chaotic systems has attracted considerable attention due to its benefits of having chaos synchronization in some engineering applications such as secure communication, chemical reactions, information processing and harmonic oscillation generation [2, 3]. It has been shown that neural networks can exhibit complicated dynamics and even chaotic behavior if the parameters and time delays are appropriately chosen for the neural networks [4,5]. Therefore, some chaotic neural networks with delays could be as models when we study the synchronization [6].

Recently, some works dealing with synchronization phenomena in delayed neural networks have also appeared, for example, see [6]-[12] and references therein. In [6]-[9], the coupled connected neural networks with delays were considered, several sufficient conditions for synchronization of such neural networks were obtained by Lyapunov stability theory and the linear matrix inequality technique. In [10]-[12], authors investigated the synchronization problem of some chaotic neural networks with delays. Using the drive-response concept, the control laws were derived to achieve the synchronization of two identical chaotic neural networks.

It is worth pointing out that, the reported works in [6]-[12] focused on synchronizing of two identical chaotic neural networks with different initial conditions. In practice, the chaotic systems are inevitably subject to some environmental

© Springer-Verlag Berlin Heidelberg 2011

changes, which may render their parameters to be variant. Furthermore, from the point of view of engineering, it is very difficult to keep the two chaotic systems to be identical all the time. Therefore, it is important to study the synchronization problem of nonidentical chaotic neural networks. Obviously, when the considered drive and response neural network are distinct and with time delay, it becomes more complex and challenging. On the study for synchronization problem of two nonidentical chaotic systems, one usually adopt adaptive control approach to establish synchronization conditions, for example, see [13] and references therein. Recently, the integral sliding mode control approach is also employed to investigate synchronization of nonidentical chaotic delayed neural networks [14,15]. In [14], an integral sliding mode control approach is proposed to address synchronization for two nonidentical chaotic neural networks with constant delay. Based on the drive-response concept and Lyapunov stability theory, both delay-independent and delay-dependent conditions in LMIs are derived under which the resulting error system is globally asymptotically stable in the specified switching surface, and a sliding mode controller is synthesized to guarantee the reachability of the specified sliding surface. In [15], the projective synchronization for two nonidentical chaotic neural networks with constant delay was investigated, a delay-dependent sufficient condition was derived by sliding mode control approach, LMI technique and Lyapunov stability theory.

However, when the considered neural networks are with time-varying delays which are not differentiable, it may be difficult to use the approaches in [14,15] to investigate synchronization of two nonidentical chaotic neural networks with time-varying delays. In addition, these results in [14,15] have conservatism to some extent due to technicality of structuring Lyapunov function, which exist room for further improvement.

Motivated by the above discussions, the objective of this paper is to presents a systematic design procedure for synchronization of two nonidentical chaotic neural networks with time-varying delays.

2 Problem Formulation and Preliminaries

In this paper, we consider the following neural network model

$$\dot{y}(t) = -C_1 y(t) + A_1 f(y(t)) + B_1 f(y(t-\tau(t))) + J_1(t), \quad t \geq 0, \tag{1}$$

where $y(t) = (y_1(t), y_2(t), \cdots, y_n(t))^T \in \mathbb{R}^n$ is the state vector of the network at time t, n corresponds to the number of neurons; $C_1 \in \mathbb{R}^{n\times n}$ is a positive diagonal matrix, $A_1, B_1 \in \mathbb{R}^{n\times n}$ are, respectively, the connection weight matrix, the delayed connection weight matrix; $f(y(t)) = (f_1(y_1(t)), f_2(y_2(t)), \cdots, f_n(y_n(t)))^T \in \mathbb{R}^n$ denotes the neuron activation at time t; $J_1(t) \in \mathbb{R}^n$ is an external input vector; $\tau(t)$ denotes the time-varying delay. It is assumed that the measured output of system (1) is dependent on the state and the delayed states with the following form:

$$w(t) = Dy(t) + Ey(t-\tau(t)), \tag{2}$$

where $w(t) \in \mathbb{R}^m$, $D, E \in \mathbb{R}^{m\times n}$ are known constant matrices.

The initial condition associated with model (1) is given by

$$y(s) = \phi(s), \qquad s \in [-\tau, 0],$$

where $\phi(s)$ is bounded and continuously differential on $[-\tau, 0]$.

We consider the system (1) as the drive system. The response system is as follows

$$\dot{z}(t) = -C_2 z(t) + A_2 g(z(t)) + B_2 g(z(t-\tau(t))) + J_2(t) + u(t), \quad t \geq 0, \quad (3)$$

with initial condition $z(s) = \varphi(s), s \in [-\tau, 0]$, where $\varphi(s)$ is bounded and continuously differential on $[-\tau, 0]$; $C_2 \in \mathbb{R}^{n\times n}$ is a positive diagonal matrix, $A_2, B_2 \in \mathbb{R}^{n\times n}$ are, respectively, the connection weight matrix, the delayed connection weight matrix; $g(z(t)) = (g_1(z_1(t)), g_2(z_2(t)), \cdots, g_n(z_n(t)))^T \in \mathbb{R}^n$ denotes the neuron activation at time t; $J_2(t) \in \mathbb{R}^n$ is an external input vector; $u(t)$ is the appropriate control input that will be designed in order to obtain a certain control objective.

Let $x(t) = y(t) - z(t)$ be the error state, then the error system can be obtained from (1) and (3) as follows

$$\begin{aligned}\dot{x}(t) = &-C_1 x(t) + A_1 h(x(t)) + B_1 h(x(t-\tau(t)))\\ &+(C_2 - C_1) z(t) - A_2 g(z(t)) - B_2 g(z(t-\tau(t)))\\ &+A_1 f(z(t)) + B_1 f(z(t-\tau(t))) - u(t) + J_1(t) - J_2(t), \end{aligned} \quad (4)$$

where $h(x(t-\tau(t))) = f(y(t-\tau(t))) - f(z(t-\tau(t)))$, and $x(s) = \phi(s) - \varphi(s), s \in [-\tau, 0]$.

Definition 1. *The drive system (1) and the response system (3) are said to be globally asymptotically synchronized, if system (4) is globally asymptotically stable.*

The aim of the paper is to design a controller $u(t)$ to let the response system (3) synchronizes with the drive system (1).

Since dynamic behavior of error system (4) relies on both error state $z(t)$ and chaotic state $z(t)$ of response system (3), complete synchronization between two nonidentical chaotic neural networks (1) and (3) cannot be achieved only by utilizing output feedback control. To overcome the difficulty, an integral sliding mode control approach will be proposed to investigate the synchronization problem of two nonidentical chaotic neural networks (1) and (3). That is, an integral sliding mode controller is designed such that the sliding motion is globally asymptotically stable, and the state trajectory of the error system (4) is globally driven onto the specified sliding surface and maintained there for all subsequent time.

To utilize the information of the measured output $w(t)$, a suitable sliding surface is constructed as

$$\begin{aligned}S(t) = x(t) + \int_0^t \Big[&C_1 x(\xi) - A_1 h(x(\xi)) - B_1 h(x(\xi - \tau(\xi)))\\ &+K\Big(w(\xi) - Dz(\xi) - Ez(\xi - \tau(\xi))\Big)\Big] d\xi, \end{aligned} \quad (5)$$

where $K \in \mathbb{R}^{n\times m}$ is a gain matrix to be determined.

It follows from (2), (4) and (5) that

$$S(t) = x(0) + \int_0^t \Big[(C_2 - C_1)z(\xi) - A_2 g(z(\xi)) - B_2 g(z(\xi - \tau(\xi))) \\ + A_1 f(z(\xi)) + B_1 f(z(\xi - \tau(\xi))) - u(\xi) + J_1(\xi) - J_2(\xi) \\ + KDx(\xi) + KEx(\xi - \tau(\xi))\Big)\Big] d\xi. \quad (6)$$

According to the sliding mode control theory [16], it is true that $S(t) = 0$ and $\dot{S}(t) = 0$ as the state trajectories of the error system (4) enter into the sliding mode. It thus follows from (6) and $\dot{S}(t) = 0$ that an equivalent control law can be designed as

$$u(t) = (C_2 - C_1)z(t) - A_2 g(z(t)) - B_2 g(z(t - \tau(t))) + A_1 f(z(t)) \\ + B_1 f(z(t - \tau(t))) + J_1(t) - J_2(t) + KDx(t) + KEx(t - \tau(t)). \quad (7)$$

Substituting (7) into (4), the sliding mode dynamics can be obtained and described by

$$\dot{x}(t) = -(C_1 + KD)x(t) - KEx(t - \tau(t)) + A_1 h(x(t)) + B_1 h(x(t - \tau(t))). (8)$$

Throughout this paper, we make the following assumptions:
(H1). There exists constant $\tau > 0$ such that

$$0 \le \tau(t) \le \tau.$$

(H2). For any $j \in \{1, 2, \cdots, n\}$, there exist constants F_j^- and F_j^+ such that

$$F_j^- \le \frac{f_j(\alpha_1) - f_j(\alpha_2)}{\alpha_1 - \alpha_2} \le F_j^+$$

for all $\alpha_1 \neq \alpha_2$.

3 Main Result

For presentation convenience, in the following, we denote

$$F_1 = \mathrm{diag}(F_1^- F_1^+, \cdots, F_n^- F_n^+), \quad F_2 = \mathrm{diag}(\frac{F_1^- + F_1^+}{2}, \cdots, \frac{F_n^- + F_n^+}{2}).$$

$$F^- = \mathrm{diag}(F_1^-, F_2^-, \cdots, F_n^-), \quad F^+ = \mathrm{diag}(F_1^+, F_2^+, \cdots, F_n^+).$$

Theorem 1. *Assume that the conditions (***H1***) and (***H2***) hold and the measured output of drive neural network (1) is condition (2). If there exist a symmetric positive definite matrix P, an inverse matrix Q, four positive diagonal matrices*

R, L, W and S, and seven matrices Y, X_{ij} $(i,j=1,2,3, i \leq j)$ such that the following two LMIs hold:

$$X = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{12}^T & X_{22} & X_{23} \\ X_{13}^T & X_{23}^T & X_{33} \end{bmatrix} > 0, \tag{9}$$

$$\Omega = \begin{bmatrix} \Omega_{11} & \Omega_{12} & \Omega_{13} & F_2 W & 0 \\ * & \Omega_{22} & -YE & \Omega_{24} & QB_1 \\ * & * & \Omega_{33} & 0 & F_2 S \\ * & * & * & -W & 0 \\ * & * & * & * & -S \end{bmatrix} < 0, \tag{10}$$

where $\Omega_{11} = \tau X_{11} + X_{13} + X_{13}^T - F_1 W$, $\Omega_{12} = P - RF^- + LF^+ - C_1 Q^T - D^T Y^T$, $\Omega_{13} = \tau X_{12} - X_{13} + X_{23}^T$, $\Omega_{22} = -Q - Q^T + \tau X_{33}$, $\Omega_{24} = R - L + QA_1$, $\Omega_{33} = \tau X_{22} - X_{23} - X_{23}^T - F_1 S$, then the response neural network (3) can globally asymptotically synchronize the drive neural network (1), and the gain matrix K can be designed as

$$K = Q^{-1} Y. \tag{11}$$

Proof. Consider the following Lyapunov-Krasovskii functional as

$$\begin{aligned} V(t) = {} & x^T(t) P x(t) + \int_{-\tau}^{0} \int_{t+\xi}^{t} \dot{x}^T(s) X_{33} \dot{x}(s) ds d\xi \\ & + 2 \sum_{i=1}^{n} r_i \int_0^{x_i(t)} (h_i(s) - F_i^- s) ds + 2 \sum_{i=1}^{n} l_i \int_0^{x_i(t)} (F_i^+ s - h_i(s)) ds \\ & + \int_0^t \int_{\xi - \tau(\xi)}^{\xi} \nu^T(\xi, s) X \nu(\xi, s) ds d\xi, \end{aligned} \tag{12}$$

where $\nu(\xi, s) = (x^T(\xi), x^T(\xi - \tau(\xi)), \dot{x}^T(s))^T$, $R = \text{diag}\{r_1, r_2, \cdots, r_n\}$, $L = \text{diag}\{l_1, l_2, \cdots, l_n\}$.

Calculating the time derivative of $V(t)$, we obtain

$$\begin{aligned} \frac{dV(t)}{dt} = {} & 2x^T(t) P \dot{x}(t) + \tau \dot{x}^T(t) X_{33} \dot{x}(t) - \int_{t-\tau}^{t} \dot{x}^T(s) X_{33} \dot{x}(s) ds \\ & + 2\dot{x}^T(t) R \Big(h(x(t)) - F^- x(t) \Big) + 2\dot{x}^T(t) L \Big(F^+ x(t) - h(x(t)) \Big) \\ & + \tau(t) \begin{pmatrix} x(t) \\ x(t-\tau(t)) \end{pmatrix}^T \begin{pmatrix} X_{11} & X_{12} \\ X_{12}^T & X_{22} \end{pmatrix} \begin{pmatrix} x(t) \\ x(t-\tau(t)) \end{pmatrix} + 2x^T(t) X_{13} x(t) \\ & - 2x^T(t) X_{13} x(t-\tau(t)) + 2x^T(t-\tau(t)) X_{23} x(t) \\ & - 2x^T(t-\tau(t)) X_{23} x(t-\tau(t)) + \int_{t-\tau(t)}^{t} \dot{x}^T(s) X_{33} \dot{x}(s) ds. \end{aligned} \tag{13}$$

From the trajectories of model (8), we have

$$0 = 2\dot{x}^T(t)Q\Big[- \dot{x} - (C_1 + KD)x(t) - KEx(t-\tau(t)) \\ +A_1h(x(t)) + B_1h(x(t-\tau(t)))\Big]. \tag{14}$$

For two positive diagonal matrices W and S, we can get from assumption (**H2**) that [17]

$$\begin{bmatrix} x(t) \\ h(x(t)) \end{bmatrix}^T \begin{bmatrix} F_1W & -F_2W \\ -F_2W & W \end{bmatrix} \begin{bmatrix} x(t) \\ h(x(t)) \end{bmatrix} \le 0. \tag{15}$$

$$\begin{bmatrix} x(t-\tau(t)) \\ h(x(t-\tau(t))) \end{bmatrix}^T \begin{bmatrix} F_1S & -F_2S \\ -F_2S & S \end{bmatrix} \begin{bmatrix} x(t-\tau(t)) \\ h(x(t-\tau(t))) \end{bmatrix} \le 0. \tag{16}$$

It follows from inequalities (13)-(16) and assumption (**H1**) that

$$\begin{aligned} \frac{dV(t)}{dt} \le\ & x^T(t)(\tau X_{11} + 2X_{13} - F_1W)x(t) \\ & +2x^T(t)(P_1 - RF^- + LF^+ - C_1Q^T - D^TK^TQ^T)\dot{x}(t) \\ & +2x^T(t)(\tau X_{12} - X_{13} + X_{23}^T)x(t-\tau(t)) + 2x^T(t)F_2Wh(x(t)) \\ & +\dot{x}^T(t)(-2Q + \tau X_{33})\dot{x}(t) - 2\dot{x}^T(t)QKEx(t-\tau(t)) \\ & +2\dot{x}^T(t)(R - L + QA_1)h(x(t)) + 2\dot{x}^T(t)QB_1h(x(t-\tau(t))) \\ & +x^T(t-\tau(t))(\tau X_{22} - 2X_{23} - F_1S)x(t-\tau(t)) \\ & +2x^T(t-\tau(t))F_2Sh(x(t-\tau(t))) - h^T(x(t))Wh(x(t)) \\ & -h^T(x(t-\tau(t)))Sh(x(t-\tau(t))) \\ =\ & \alpha^T(t)\Omega\alpha(t). \end{aligned} \tag{17}$$

From (10) and (17), we know that the error dynamical system (8) is globally asymptotically stable by the Lyapunov stability theory. Accordingly, the response neural network (3) can globally asymptotically synchronize the drive neural network (1). The proof is completed.

4 An Example

In the drive neural network (1), we set

$$C_1 = \begin{bmatrix} 1.1 & 0 \\ 0 & 1 \end{bmatrix}, A_1 = \begin{bmatrix} 2.5 & -0.1 \\ -5.2 & 3.2 \end{bmatrix}, B_1 = \begin{bmatrix} -1.6 & -0.2 \\ -0.3 & -2.5 \end{bmatrix}, J_1 = \begin{bmatrix} -0.8\cos(2t) \\ 3\sin(0.1t) \end{bmatrix},$$

$$f_1(y) = f_2(y) = \tanh(y), \tau(t) = 1 + 0.1|\sin t|.$$

Then the dynamical behavior of neural network (1) with initial condition $y_1(s) = -0.03$, $y_2(s) = -0.1$, $s \in [-1.1, 0]$ exhibits a chaotic behavior as shown Fig. 1.

The parameters of the measured output (2) are given as

$$D = \begin{bmatrix} 0.5 & 0 \\ 0.1 & -0.1 \end{bmatrix}, E = \begin{bmatrix} 0.9 & 0 \\ 0.1 & 0 \end{bmatrix}.$$

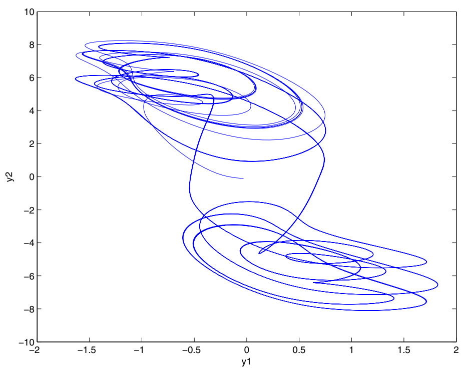


Fig. 1. Chaotic behavior of system (1)

The response neural network (3) is considered as

$$C_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1.2 \end{bmatrix}, A_2 = \begin{bmatrix} 1+\frac{\pi}{4} & 20 \\ 0.1 & 1+\frac{\pi}{4} \end{bmatrix}, B_2 = \begin{bmatrix} -1.3\sqrt{2}\pi & 0.1 \\ 0.1 & -1.3\sqrt{2}\pi \end{bmatrix},$$

$$J_2 = \begin{bmatrix} 0.1\sin(2t) \\ -0.023\cos(4t) \end{bmatrix}, g_1(z) = g_2(z) = \frac{1}{2}(|z+1| - |z-1|).$$

Then the dynamical behavior of neural network (3) with initial condition $z_1(s) = 0.02$, $z_2(s) = 0.03$, $s \in [-1.1, 0]$ exhibits a chaotic behavior as shown Fig. 2.

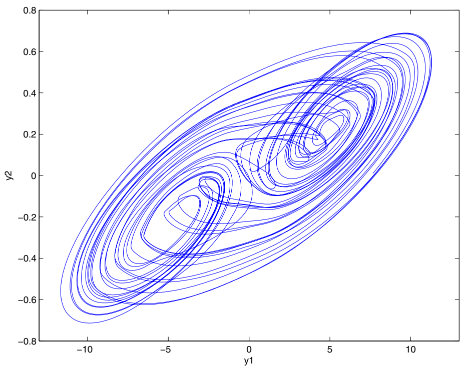


Fig. 2. Chaotic behavior of system (3) without the controller $u(t)$

It can be verified that assumptions (**H1**) and (**H2**) are satisfied, and $F_1 = 0$, $F_2 = \mathrm{diag}\{0.5, 0.5\}$, $F^- = 0$, $F^+ = \mathrm{diag}\{1, 1\}$.

By the Matlab LMI Control Toolbox, we find a solution to the LMIs in (8) and (9), and obtain the gain matrix K as

$$K = 10^7 \begin{bmatrix} -0.0010 & 0.0086 \\ -0.1672 & 1.5048 \end{bmatrix}.$$

By Theorem 1, we know that the response neural network (3) can globally asymptotically synchronize the drive neural network (1).

5 Conclusions

In this paper, the synchronization of two nonidentical chaotic neural networks with time-varying delays has been investigated. By use of the sliding mode control theory, and employing a combination of the free-weighting matrix method, inequality technique and Lyapunov theory, a sliding mode controller has been designed to achieve the asymptotical synchronization of two nonidentical chaotic neural networks with time-varying delays. The provided condition is expressed in terms of linear matrix inequalities (LMIs), and is dependent on the time delay. A simulation example has been also given to show the effectiveness of the obtained result.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 60974132 and 10772152.

References

1. Pecora, L.M., Carroll, T.L.: Synchronization in Chaotic Systems. Physical Review Letters 64, 821–824 (1990)
2. Boccaletti, S., Kurths, J., Osipov, G., Valladares, D.L., Zhou, C.S.: The Synchronization of Chaotic Systems. Physics Reports 366, 1–101 (2002)
3. Salarieh, H., Alasty, A.: Adaptive Synchronization of Two Chaotic Systems with Stochastic Unknown Parameters. Communications in Nonlinear Science and Numerical Simulation 14, 508–519 (2009)
4. Gilli, M.: Strange Attractors in Delayed Cellular Neural Networks. IEEE Transactions on Circuits and Systems I 40, 849–853 (1993)
5. Lu, H.T.: Chaotic Aattractors in Delayed Neural Networks. Physics Letters A 298, 109–116 (2002)
6. Lu, W.L., Chen, T.P.: Synchronization of Coupled Connected Neural Networks with Delays. IEEE Transactions on Circuits and Systems I 51, 2491–2503 (2004)
7. Liu, Y.R., Wang, Z.D., Liu, X.H.: On Synchronization of Coupled Neural Networks with Discrete and Unbounded Distributed Delays. International Journal of Computer Mathematics 85, 1299–1313 (2008)
8. Lu, J.Q., Ho, D.W.C., Cao, J.D.: Synchronization in An Array of Nonlinearly Coupled Chaotic Neural Networks with Delay Coupling. International Journal of Bifurcation and Chaos 18, 3101–3111 (2008)
9. Cao, J.D., Li, L.L.: Cluster Synchronization in An Array of Hybrid Coupled Neural Networks with Delay. Neural Networks 22, 335–342 (2009)
10. Yu, W.W., Cao, J.D.: Synchronization Control of Stochastic Delayed Neural Networks. Physica A 373, 252–260 (2007)
11. Park, J.H.: Synchronization of Cellular Neural Networks of Neutral Type Via Dynamic Feedback Controller. Chaos, Solitons and Fractals 42, 1299–1304 (2009)

12. Karimi, H.R., Maass, P.: Delay-Range-Dependent Exponential H_∞ Synchronization of A Class of Delayed Neural Networks. Chaos, Solitons and Fractals 41, 1125–1135 (2009)
13. Zhang, H.G., Xie, Y.H., Wang, Z.L., Zheng, C.D.: Adaptive Synchronization Between Two Different Chaotic Neural Networks with Time Delay. IEEE Transactions on Neural Networks 18, 1841–1845 (2007)
14. Huang, H., Feng, G.: Synchronization of Nonidentical Chaotic Neural Networks with Time Delays. Neural Networks 22, 869–874 (2009)
15. Zhang, D., Xu, J.: Projective Synchronization of Different Chaotic Time-Delayed Neural Networks Based on Integral Sliding Mode Controller. Applied Mathematics and Computation 217, 164–174 (2010)
16. Utkin, V.I.: Sliding Modes in Control and Optimization. Springer, Berlin (1992)
17. 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)

Blow-Up for a Class of Parabolic Equations with Nonlinear Boundary Conditions

Leina Zhao

Department of Mathematics, Chongqing Jiaotong University,
Chongqing 400074, China
zhaoleina3529@tom.com

Abstract. In this paper, we consider the following equation

$$\frac{\partial h(u)}{\partial t} = \Delta u + a(x,t)f(u), \quad \Omega \times (0,T),$$

with initial condition and third boundary condition. By constructing an auxiliary function and using maximum principles, we established a sufficient conditions for the blow-up of solutions. The blow-up rate and the blow-up set were also considered under appropriate assumption. This result generalizes and improves earlier results in literatures.

Keywords: Blow up, Blow-up rate, Blow-up set.

1 Introduction

In this work, we study the following equation

$$\frac{\partial h(u)}{\partial t} = \Delta u + a(x,t)f(u), \quad \text{in} \quad \Omega \times (0,T) \tag{1}$$

$$\frac{\partial u}{\partial n} + b(x,t)g(u) = 0, \quad \text{on} \quad \partial\Omega \times (0,T) \tag{2}$$

$$u(x,0) = u_0(x), \quad \text{in} \quad \Omega, \tag{3}$$

where $\Omega \subset R^N$ is a bounded domain with smooth boundary $\partial\Omega$; $\partial/\partial n$ represents the outward normal derivative on $\partial\Omega$; $a(x,t)$ and $b(x,t)$ are nonnegative functions. For $s > 0$, $h(s)$, $f(s)$, $g(s)$ are positive and increasing functions.

The problem of blow-up solution for nonlinear parabolic equations with different boundary conditions has been investigated extensively by many authors (see[1-14]). For example, Imai and Mochizuki [8] studied the following problem:

$$\begin{cases} (h'(u))_t = \Delta u + f(u), \quad \text{in} \quad \Omega \times (0,T) \\ \frac{\partial u}{\partial n} = 0, \quad \text{on} \quad \partial\Omega \times (0,T) \\ u(x,0) = u_0(x) > 0, \quad \text{in} \quad \Omega, \end{cases} \tag{4}$$

where Ω is a bounded domain of R^N with smooth boundary. Ding and Gao [2] studied the problem

$$\begin{cases} (h(u))_t = \nabla \cdot (a(u,t)b(x)\nabla u) + g(t)f(u), \quad \text{in} \quad \Omega \times (0,T) \\ \frac{\partial u}{\partial n} = 0, \quad \text{on} \quad \partial\Omega \times (0,T) \\ u(x,0) = u_0(x) > 0, \quad \text{in} \quad \Omega, \end{cases} \tag{5}$$

© Springer-Verlag Berlin Heidelberg 2011

where Ω is a bounded domain of R^N with smooth boundary. The sufficient conditions were obtained there for the global solution and blow-up solution. Meanwhile, the upper estimate of "blow-up rate" were also given.

In this paper, we main consider the blow-up results to problem (1)-(3). In the last decades, some special cases of the problem had been considered, and the reader is referred to [4, 7] and the references therein. By constructing an auxiliary function and using maximum principles, we established a sufficient conditions for the blow-up of solutions. The blow-up rate and the blow-up set were also considered under appropriate assumption. Our main results are as follows.

Theorem 1. *Let $a_t(x,t) \geq 0$, $b_t(x,t) \leq 0$, $a(x,t) \geq a_0 > 0$. $\Delta u_0(x) + a(x,0)$ $f(u_0(x)) > 0$. Suppose that there exist a function $F(s)$ and a positive constant A such that*

$$F(s) \geq 0, \quad F'(s) \geq 0, \quad F''(s) \geq 0 \quad for \quad s > 0 \tag{6}$$

$$f'(s)F(s) - F'(s)f(s) - h''(s)F^2(s) \geq 0 \quad for \quad s > 0 \tag{7}$$

$$F'(s)g(s) - F(s)g'(s) \geq 0 \quad for \quad s \geq A \ and \int_1^\infty \frac{ds}{F(s)} < \infty, \tag{8}$$

Then the solution of (1)-(3) blows up in a finite time and there exists a positive constant δ such that

$$\sup_{x\in\bar{\Omega}} u(x,t) \leq H(\delta(T-t)), \tag{9}$$

where $H(s)$ is the inverse function of $G(s) = \int_s^\infty \frac{dy}{F(y)}$.

Theorem 2. *If the above conditions in Theorem 1 are satisfied, and additional suppose that $0 < a_0 \leq a(x,t) \leq \ a_1 < \infty$, and $G_0(s) = \int_1^\infty \frac{h'(y)dy}{f(y)} < +\infty$. We deduce that*

$$\sup_{x\in\bar{\Omega}} u(x,t) \geq H_0(a_1(T-t)).$$

where H_0 is the inverse function of $G_0(s) = \int_s^\infty \frac{h'(y)dy}{f(y)}$.

We give special case of problem (1)-(3) where $h(u) = u^m$, $f(u) = u^p$, $g(u) = u^q$, and the other conditions in Theorem 1and Theorem 2 are satisfied. Then we have the following corollary.

Corollary 1. *If u is the solution of the following problem*

$$\frac{\partial u^m}{\partial t} = \Delta u + a(x,t)u^p, \quad in \quad \Omega \times (0,T)$$
$$\frac{\partial u}{\partial n} + b(x,t)u^q = 0, \quad on \quad \partial\Omega \times (0,T)$$
$$u(x,0) = u_0(x), \quad in \quad \Omega,$$

where $m, p, q > 0$, $p > m+1$ and $p > m+q-1$, then $u(x,t)$ must blows up in finite time T, and there is a constant c_0 and c_1 such that

$$\frac{C_0}{(T-t)^{1/1(p-m)}} \le u(x,t) \le \frac{C_0}{(T-t)^{1/(p-m)}}.$$

Friedam et al. in [5] discussed the blow-up set for a special case of problem (1)-(3). By using their method, we also obtain the same result to the general problem (1)-(3). For this, We consider the case when the initial value is radially symmetric and radially decreasing in a spherical domain $\Omega = B(R) = \{|x| \le R\} \subset R^n$. And we also suppose $a(x,t) = a(t)$, $b(x,t) = b(t)$ with $a_t \ge 0$, $b_t \le 0$, and the initial value $u_0(x)$ is a radially symmetric function, and $u'(0) = u'(R) + b(0)g(u_0(R)) = 0$. By the uniqueness, we see that the solution is also radially symmetric and can be written as $u(r,t)$. For the nonlinear term f we assume in addition the following condition (F_1): There ia a function $F = F(u)$ such that

(i) $F(s) \ge 0,\ F'(s) \ge 0 \quad \text{and} \quad F''(s) \ge 0 \quad \text{for} \quad s > 0$
(ii) $\int_1^\infty \frac{ds}{F(s)} < +\infty$
(iii) for each (large) $U > 0$, there is a constant $c = c(U) > 0$ such that

$$f'(s)F(s) - f(s)F'(s) \ge cF(s)F'(s), \quad \text{for } s \ge U.$$

Theorem 3. *Assume that*

$$h(s) \ge 0,\ h'(s) \ge 0,\ h''(s) \le 0,\ \textit{for } s > 0 \tag{10}$$

and

$$u_0(r) > 0,\ u_0'(r) \le 0,\ u_0'(r) \ne 0,\ \textit{for } r \in (0,R) \tag{11}$$
$$\Delta u_0 + a(x,0)f(u_0) > 0. \tag{12}$$

Let the solution blows up in a finite time, then its blow up set is consist of single point $x = 0$.

Remark 1. It from (12) and the maximum principle that $u_t \ge 0$, in $(0,t) \times \Omega$.

Remark 2. If the condition F_1 is satisfied, then for each $M > 0$, there is a constant $C = C(M) > 0$, such that

$$f'(s)F(s) - f(s)F'(s) \ge MF(s), \quad s > C. \tag{13}$$

By (i) and (ii) of F_1, we have $F'(s) \to \infty$ as $s \to \infty$. Thus, for fixed $U > 0$ and $c > 0$ satisfying (iii), we can get $cF'(s) \ge M$, $s \ge C$ for a sufficiently large constant $C = C(M) \ge c(U)$, which yields (13).

The rest of this paper is organized as follows. In section 2, we proof Theorem 1and Theorem 2. The proof of Theorem 3 is presented in Section 3.

2 Blow-Up Conditions

Proof. It follows from $u_0(x) > 0$ in Ω and the maximum principle that $u(x,t) \geq 0$, in $\Omega \times (0,T)$. Let $w = u_t$, since $w(x,0) = 1/(h'(u_0))(\Delta u_0(x) + a(x,0)f(u_0(x))) > 0$, $a_t(x,t) \geq 0$, $b_t(x,t) \leq 0$, we have that

$$\begin{cases} (h'(u)w)_t - \Delta w \geq a(x,t)f'(u)w, & \text{in } \Omega \times (0,T) \\ \frac{\partial w}{\partial n} + b(x,t)g'(u)w \geq 0, & \text{on } \partial\Omega \times (0,T) \\ w(x,0) > 0, & \text{in } \Omega \end{cases} \tag{14}$$

Then by the maximum principle, there exists a constants C and $\varepsilon_0 > 0$ such that

$$w(x,t) = u_t(x,t) \geq C, \quad \text{in} \quad \Omega \times (\varepsilon_0, T)$$

Next we introduce the following function

$$J(x,t) = u_t - \delta F(u) \tag{15}$$

where δ will be determined later. At first, we assume $\delta < \inf\{a_0, C/F(A)\}$. After computation we have that

$$\begin{aligned}
&(h'(u)J)_t - \Delta J \\
&\quad = ((h(u))_t - \Delta u)_t - \delta F'((h(u)_t) - \Delta u) + \delta''|\nabla u|^2 - \delta h''(u)F(u)u_t \\
&\quad = [a(x,t)f'(u) - \delta h''(u)F(u)]J + a(x,t)\delta[f'(u)F(u) - F'(u)f(u)] \\
&\qquad + \delta F''(u)|\nabla u|^2 - \delta^2 h''(u)F^2(u) \\
&\quad \geq \delta[f'(u) - h''(u)F(u)]J + \delta^2[f'(u)F(u) - F'(u)f(u) - h''(u)F^2(u)] \\
&\qquad + \delta F''(u)|\nabla u|^2 \\
&\quad \geq \delta[f(u) - h''(u)F(u)]J
\end{aligned} \tag{16}$$

Thus

$$(h'(u)J)_t - \Delta J - \delta[f(u) - h''(u)F(u)]J \geq 0 \tag{17}$$

We next take δ so small such that

$$J(x,\varepsilon_0) > 0, \quad \text{in } \Omega \tag{18}$$

We now want to show that

$$J(x,\varepsilon_0) > 0, \quad \text{in } \bar{\Omega} \times (\varepsilon_0, T) \tag{19}$$

Suppose that J admits a negative minimum at a point $(x_0,t_0) \in \bar{\Omega} \times (\varepsilon_0,T)$. By the maximum principle $(x_0,t_0) \in \partial\Omega \times (\varepsilon_0,T)$. At first, suppose $u(x_0,t_0) \leq A$, since that F is an increasing function, we have

$$J(x_0,t_0) \geq C - \delta F(A) \tag{20}$$

Since $\delta < C/F(A)$, we get $J(x_0,t_0) > 0$, which is a contradiction. Now suppose that $u(x_0,t_0) > A$, the maximum principle implies that

$$\frac{\partial J}{\partial n} + b(x,t)g'(u)J < 0, \quad \text{at} \quad (x_0,t_0) \tag{21}$$

Therefor we have

$$\frac{\partial u_t}{\partial n} + b(x,t)g'(u)u_t - \delta\left[\frac{\partial F(u)}{\partial n} + b(x,t)g'(u)F(u)\right] < 0, \quad \text{at} \quad (x_0,t_0) \tag{22}$$

By (2) and (22) we have that

$$\frac{\partial F(u)}{\partial n} + b(x,t)g'(u)F(u) > 0$$

And the same time we also have $\frac{\partial F(u)}{\partial n} = F'(u)\frac{\partial u}{\partial n} = -F'(u)b(x,t)g(u)$ Thus we have $g'(u)F(u) - F'(u)g(u) > 0$, which contradicts to (8). Therefor we deduce that

$$u_t > \delta F(u), \quad \text{in} \quad \Omega \times (\varepsilon_0, T) \tag{23}$$

That is to say

$$-(G(u))_t = \frac{u_t}{F(u)} \geq \delta \tag{24}$$

Integrating (24) over (ε_0, T), we have

$$G(u(x,\varepsilon_0)) \geq G(u(x,\varepsilon_0)) - G(u(x,T)) \geq \delta(T - \varepsilon_0) \tag{25}$$

Therefor T is finite and u blows up in a finite time. Integrating again (25) over (t,T) we see that

$$G(u(x,t)) \geq \delta(T-t) \tag{26}$$

By (8) we obtain that

$$u(x,t) \leq H[\delta(T-t)].$$

Proof. In the additional condition, $0 < a_0 \leq \delta_0 \leq a_1 < \infty$ and $G_0(s) = \int_1^\infty \frac{h'(s)ds}{f(s)} < +\infty$, we will prove the lower bound of the blow-up rate. Put $U(t) = \sup_{x\in\bar{G}} u(x,t)$, since $\bar{\Omega}$ is compact, there exist $x_i \in \bar{\Omega}(i=1,2)$ such that $u(t_i) = u(x_i,t_i)$ for $t_i \geq 0$.

$$\begin{aligned} h(U(t_2)) - h(U(t_1)) &\leq h(u(x_2,t_2)) - h(u(x_2,t_1)) \\ &= (t_2-t_1)(h(U(x_2,t_2)))_t + o(t_2-t_1) \end{aligned} \tag{27}$$

therefor we have

$$\frac{h(U(t_2)) - h(U(t_1))}{t_2 - t_1} \leq (h(u(x_2,t_2)))_t + \mathrm{o}(1) \tag{28}$$

We also have that

$$(h(u(x_2,t_2)))_t \leq a(x_2,t_2)f(u(x_2,t_2)) = a(x_2,t_2)f(U(t_2)) \leq a_1 f(U(t_2)) \tag{29}$$

Because $\Delta u(x_2,t_2) \leq 0$, $a(x_2,t_2) \leq \delta_1$. Since

$$\lim_{t_2\to t_1} \frac{h(U(t_2)) - h(U(t_1))}{t_2 - t_1} = (h(U(t_1)))_t \tag{30}$$

it follows from (28) and (30) that $(h(U(t_1)))_t \leq a_1 f(U(t_1))$. Consequently, we deduce that

$$\sup_{x\in\bar{\Omega}} u(x,t) \geq H_0(a_1(T-t)),$$

where H_0 is the inverse function of $G_0(s) = \int_s^\infty \frac{h'(y)dy}{f(y)}$. The proof of Theorem 2 is complete.

3 Blow-Up Set

Lemma 1. *Suppose that*

$$u_0(r) > 0, \quad u_0'(r) \leq 0, \quad u_0'(r) \neq 0 \textit{ for } r \in (0,R),$$

Then we have that

$$u(r,t) > 0, \textit{ for } (r,t) \in (0,t) \times [0,R] \tag{31}$$

and

$$u_r(r,t) < 0, \textit{ for } (r,t) \in (0,t) \times (0,R). \tag{32}$$

Proof. The assertion (31) is obtained by the maximum principle and the strong maximum principle. And the assertion (32) can be shown in a similar way with the principles applied to the function $v = u_r$, a solution of

$$\begin{cases} (h'(u)v)_t = v_{rr} + \frac{n-1}{r}v_r - \frac{n-1}{r^2}v + a(t)f'(u)v, \quad (t,r) \in (0,T) \times (0,R) \\ v(0,t) = v_r(R,t) + b(t)g'(u(R,t))v(R,t) = 0, \quad t \in (0,T) \\ v(r,0) = u_0'(r), \quad r \in [0,R]. \end{cases} \tag{33}$$

We have only to show that there is no blow-up point in $\Omega \setminus 0$, if the assertion is false, there would be a blow-up point x_0 in $\overline{\Omega}$ such that $x_0 \neq 0$, $r_0 = |x_0| \in (0,R]$. In virtue of Lemma (1) we have

$$\lim_{t\to T} u(t,r) = +\infty, \quad \text{for all} \quad r \in [0,r_0)$$

Let y and z be fixed number such that $0 < y < z < r_0/\sqrt{N}$.

Consider the auxiliary function

$$J(x,t) = u_{x_1}(x,t) + c(x)F(u(t,x)) \tag{34}$$

with

$$c(x) = \varepsilon \prod_{j=1}^{N} \sin(\mu(x_j - y)) \tag{35}$$

where $\varepsilon > 0$ is to be chosen later, and $\mu = \pi/(z-y)$. Putting $G = \{x \in \Omega; y < x_j < z, j = 1,\cdots,N\}$. We show that there is a time $\tau \in (0,t)$ sufficiently close to T and a sufficiently small constant $\varepsilon > 0$, such that

$$-u_{x_1} \geq c(x)F(u), \quad \text{for} \quad (t,x) \in (\tau,T) \times G \tag{36}$$

Using Lemma 1, we see that $J = J(t, x)$ satisfies

$$u_{x_1} < 0, \quad \text{for} \quad (t, x) \in (\tau, T) \times \partial G \tag{37}$$

Since $u_{x_1} = |\nabla u| \cos(e_1, -r) < 0$, where e_1 and r are unit vectors of the positive x_1 direction, and the radial direction, respectively.

On the other hand, we have

$$(h'(u)J)_t = h''(u)u_{x_1}u_t + ch''(u)F(u)u_t + h'(u)u_{tx_1} + c(x)h'(u)F'(u)u_t, \tag{38}$$

$$\triangle J = \triangle u_{x_1} + F(y)\triangle c + 2F'(\nabla c \cdot \nabla u) + cF''|\nabla u|^2 + cF'\triangle u \tag{39}$$

Thus we have that

$$\begin{aligned}
&(h'(u)J)_t - \triangle J \\
&\quad = ((h(u))_t - \triangle u)_{x_1} + cF'((h(u))_t - \triangle u) + cFh''(u)u_t - F\triangle c \\
&\qquad - 2F'(\nabla c \cdot \nabla u) - cF''|\nabla u|^2 \\
&\quad = a(t)f'(u)(J - cF) + cF'a(t)f(u) + cFh''u_t - 2F'A(t,x)(J - cF) \\
&\qquad + n\mu^2 cF - cF''|\nabla u|^2
\end{aligned} \tag{40}$$

where $A(t, x) = \frac{2(\nabla c \cdot \nabla u)}{u_{x_1}}$ is a bounded function subject to $A(t, x) \leq \varepsilon c_1$, in $(0, T) \times \bar{G}$ for some constant $c_1 > 0$. This can be obtained by $|\nabla c| \leq \varepsilon\mu$, $|\nabla u| = |u_{x_1}|/|\cos(e_1, -r)| = \frac{r|u_{x_1}|}{x_1}$ and $0 < r/x_1 < z/y$,for $(x, t) \in (0, T) \times \bar{G}$. Thus we get that

$$\begin{aligned}
&(h'(u)J)_t - \triangle J \\
&\quad = (a(t)f'(u) - 2F'A)J - a(t)cf'F + a(t)cF'f + cFh''u_t + 2F'AcF \\
&\qquad + n\mu^2 cF - cF''|\nabla u|^2
\end{aligned} \tag{41}$$

Then

$$\begin{aligned}
(h'(u)J)_t - \Delta u - bJ &\leq -a(t)f'cF + a(t)cF'f + 2F'AcF + n\mu^2 F \\
&\leq -c[a(t)f'F - a(t)F'f + 2F'AF + n\mu^2 F]
\end{aligned} \tag{42}$$

where $b = b(t, x) = a(t)f' + 2F'A$, and b is continuous and locally bounded in $(0, t) \times \bar{G}$. By Remark (2) there exists a constant $c = c(2n\mu^2) > 0$, such that

$$a(t)f'F - a(t)F'f \geq 2n\mu^2 F \tag{43}$$

for $\mu > c = c(2n\mu^2) > 0$, By remark (1), there is a time $\tau \in (0, T)$, such that

$$u(x, \tau) \geq c(2n\mu^2), \text{ in } \quad G \tag{44}$$

It follows from (43) that

$$u(x, t) \geq c = c(2n\mu^2), \quad \text{in} \quad G \times (0, T) \tag{45}$$

Then we deuce that

$$a(t)f'F - a(t)F'f \geq 2n\mu^2 F \tag{46}$$

On the other hand, if $\varepsilon_1 > 0$ is so small that, $c(u(\tau, z')) > 2\varepsilon\bar{c}$ in (iii) of (F_1) where $Z' = (z, ..., z)$. Then $0 < \varepsilon < \varepsilon_1$ implies

$$a(t)f'F - a(t)F'f \geq 2\varepsilon\tilde{c}F'(u)F(u), \quad \text{for} \quad (t, x) \in (\tau, T) \times G$$

Since $u(t, x) \geq u(t, z')$ for $(t, x) \in [\tau, T) \times \bar{G}$

By (41) and (46), we see

$$(h'(u)J)_t - \Delta u - bJ \leq 0, \quad \text{in} \quad (\tau, T) \times G \tag{47}$$

when $t = \tau$, it can be seen that

$$J(x, \tau) \leq \max_{x \in \bar{G}} u_{x_1}(\tau, x) + \varepsilon F(\max_{x \in \bar{G}} u(r, x)) < 0, \quad \text{for} \quad x \in \bar{G}$$

provide $0 < \varepsilon < \varepsilon_2$ for ε_2 so small. By the maximum principle we deduce that $J(x, t) < 0$, $(t, x) \in (\tau, T) \times G$. Thus

$$-\frac{u_{x_1}}{F(u)} \geq c(x)$$

Then putting $I(v) = \int_v^\infty \frac{ds}{f(s)}$ for $v > 0$. By (ii) of (F_1), we see that $0 < I(v) < \infty$ and

$$\lim_{v \to \infty} I(v) \to 0$$

Take $a = (a_1, ..., a_N) \in \partial\partial G, b = (b_1, ..., b_N) \in \partial G$, such that $a_1 = y$, $b_1 = z$, and $a_i = b_i \in (y, z)$ for $i = 2, ..., N$, by integration of $u_{x_1}/F(u) \geq c(x)$ on the line segment connecting a and b, we see that

$$I(u(t, a)) > I(u(t, a)) - I(u(t, b)) \geq \varepsilon \int_y^z c(x_1, a_2, ..., a_N)dx_1 > 0$$

where the right-hand side is independent of t. However, $t \to T$ leads to $u(t, a) \to +\infty$, and thus $I(u(t, a)) \to 0$, a contradiction witch complete the proof of Theorem 3.

References

1. Cheng, T., Zheng, G.F.: Some blow-up problems for a semilinear parabolic equation with a potential. J. Differential Equations 244, 766–802 (2008)
2. Ding, J.T., Gao, X.Y., Li, S.J.: Global existence and blow up problems for reaction diffusion model with multiple nonlinearities. J. Math. Anal. Appl. 343, 159–169 (2008)
3. Ding, J.T., Li, S.J.: Blow-up solutions and global solutions for a class of quasilinear parabolic equations with Robin boundary conditions. Comp. Math. Appl. 49, 689–701 (2005)

4. Deng, K., Levine, H.A.: The role of critical exponents in blow-up theorems: the sequel. J. Math. Anal. Appl. 243, 85–126 (2000)
5. Friedman, A., McLeod, J.B.: Blow-up of positive Solutions of Semilinear Heat Equations. Indiana Univ. Math. J. 34, 425–447 (1985)
6. Fujishima, Y., Ishige, K.: Blow-up set for a semilinear heat equation with small diffusion. Journal of Differential Equations 249, 1056–1077 (2010)
7. Galaktionov, V.A., Váquez, J.L.: The problem of blow-up in nonlinear parabolic equations. Discrete Contin. Dyn. Syst. 8, 399–433 (2002)
8. Imai, T., Mochizuki, K.: On the blow-up of solutions for quasilinear degenerate parabolic equations. Publ. Res. Inst. Math. Sci. 27, 695–709 (1991)
9. Ishige, K., Mizoguchi, N.: Blow-up behavior for semilinear heat equations with boundary conditions. Differential Integral Equations 16, 663–690 (2003)
10. Jazar, M., Kiwan, R.: Blow-up of a non-local semilinear parabolic equation with Neumann boundary conditions. Ann. Inst. H. Poincaré Anal. Non Linéaire 25, 215–218 (2008)
11. Lair, A.V., Oxley, M.E.: A necessary and sufficient condition for global existence for degenerate parabolic boundary value problem. J. Math. Anal. Appl. 221, 338–348 (1998)
12. Levine, H.A.: The role of critical exponents in blow-up theorems. SIAM Rev. 32, 262–288 (1990)
13. Qi, Y.W.: The critical exponent of parabolic equations and blow-up in R^3. Proc. Roy. Soc. Edinburgh Sect. A 128, 123–136 (1998)
14. Wang, J., Wang, Z.J., Yin, J.X.: A class of degenerate diffusion equations with mixed boundary conditions. J. Math. Anal. Appl. 298, 589–603 (2004)
15. Souplet, P.: Single-point blow-up for a semilinear parabolic system. Journal of the European Mathematical Society 11, 169–188 (2009)
16. Zhang, H.L.: Blow-up solutions and global solutions for nonlinear parabolic problems. Nonlinear Anal. 69, 4567–4575 (2008)
17. Zhang, H.L., Liu, Z.R.: Blow up of positive solution of quasilinear parabolic equations with nonlinear neuman boundary conditions. Global Journal of Pure and Applied Mathematics 2, 225–233 (2005)
18. Zhang, L.L.: Blow-up of solutions for a class of nonlinear parabolic equations. Z. Anal. Anwend. 25, 479–486 (2006)

Cluster Synchronization for Discrete-Time Complex Networks

Huiwei Wang[1] and Qiankun Song[2]

[1] College of Information Science & Engineering, Chongqing Jiaotong University, Chongqing 400074, China
[2] Department of Mathematics, Chongqing Jiaotong University, Chongqing 400074, China
qiankunsong@163.com

Abstract. This paper investigates cluster synchronization in discrete-time complex networks with both discrete and distributed time-varying delays. By utilizing a special coupling matrix and the Kronecker product, it is shown that the addressed discrete-time complex networks is achieved cluster synchronization if certain linear matrix inequality (LMI) is feasible. Finally, an example is given to demonstrate the effectiveness of the proposed criterion.

Keywords: Cluster synchronization, Complex networks, Discrete-time, Discrete delays, Distributed delays.

1 Introduction

Since the seminal works of Pecora and Carroll [1], chaos synchronization has received a great deal of interest from various fields. Synchronization phenomena in coupled chaotic systems have been extensively studied in electronic circuits, laser systems, pairs of neurons, and biological systems [2]. Cluster synchronization, as a particular synchronization phenomenon, was firstly considered for coupled chaotic oscillators, which is observed that synchronization occurs in each group, but there is no synchronization among the different groups [3]. Recently, cluster synchronization has become one of the hottest topics of discussion because it is considered to be more momentous than other synchronization types in brain science and engineering, social science, and distributed computation. The cluster synchronization criteria were proposed in [4] for coupled Josephson equations by constructing different coupling schemes. In [5], a coupling scheme with cooperative and competitive weight-couplings was used to realize cluster synchronization for connected chaotic networks. In [6], cluster synchronization of linearly coupled complex networks has been investigated under a adaptive strategy, whereas the similar topic has been addressed in [7] by employing a pinning controller. In [8], the authors dealt with cluster synchronization analysis problem in an array of hybrid coupled neural networks with constant delays. For more studies concerning cluster synchronization, please see [9] and the references therein.

© Springer-Verlag Berlin Heidelberg 2011

In 1998, Watts and Strogatz presented a simple model of network structure, which is known as small-world networks, during one decade we have witnessed the evolution of the field of complex networks [10]. It should be pointed out that most literatures on the dynamical behaviors of complex networks are concerned with continuous-time case. However, the discretization process of a continuous-time network cannot preserve the dynamics of the continuous-time part even for small sampling periods [11]. In addition, a discrete-time network is in a better position to model digitally transmitted signals in a dynamical way than its continuous-time analog. Very recently, the synchronization problem for discrete-time networks has received some initial research interests. For example, an array of coupled discrete-time neural networks in [12], as a special case of coupled complex networks, was concerned with the robust synchronization analysis. In [13], the problem of stochastic synchronization analysis is investigated for a new array of coupled discrete-time stochastic complex networks with randomly occurred nonlinearities and time delays. In [14], synchronization and state estimation is studied for an array of coupled complex discrete-time networks with the simultaneous presence of both the discrete and distributed time delays. So far, to the best of the authors' knowledge, few authors have considered the problem on cluster synchronization for discrete-time complex networks with mixed delays. Therefore, the study on cluster synchronization of discrete-time complex networks with mixed delays is not only important but also necessary.

Motivated by the above discussions, the objective of this paper is to concern with the cluster synchronization problem for discrete-time complex networks with both discrete and distributed delays. By utilizing a special coupling matrix and the Kronecker product, a new criterion is developed to achieve cluster synchronization for concerned model. Finally, a simulation example is used to demonstrate the usefulness of the proposed design methods.

2 Model Description and Preliminaries

In this paper, we consider the following discrete-time delayed neural networks consisting N coupled nodes of the form:

$$\begin{aligned} x_i(k+1) &= f(x_i(k)) + g(x_i(k-d(k))) \\ &\quad + \sum_{m=1}^{+\infty} \mu_m h(x_i(k-m)) + \sum_{j=1}^{N} G_{ij} \Gamma x_j(k), \quad i = 1, 2, \cdots, N \end{aligned} \tag{1}$$

where $x_i(k) = (x_{i1}(k), x_{i2}(k), \cdots, x_{in}(k))^T$ is the state vector of the ith node; $f(\cdot)$, $g(\cdot)$ and $h(\cdot)$ are nonlinear vector-valued functions satisfying certain conditions given later; The positive integer $d(k)$ denotes the discrete time-varying delay satisfying $0 \leq d_m \leq d(k) \leq d_M$, $k \in \mathbb{N}$, d_m and d_M are known positive integers; The constants $\mu_m \geq 0 (m = 1, 2, \cdots)$ satisfy the following convergent conditions:

$$\sum_{m=1}^{+\infty} \mu_m < +\infty, \quad \sum_{m=1}^{+\infty} m\mu_m < +\infty. \tag{2}$$

$\Gamma = \mathrm{diag}\{\gamma_1, \gamma_2, \cdots, \gamma_n\} \geq 0$ is a matrix linking the jth state variable if $\gamma_j \neq 0$. $G = G_{ij} \in \mathbb{R}^{N \times N}$ is the coupled configuration matrix of the network with $G_{ij} \geq 0 (i \neq j)$, but not all zero. Here, the coupling configuration matrix may not identical and satisfies:

$$G_{ii} = \sum_{j=1, j\neq i}^{N} G_{ij}, \quad G_{ij} \geq 0 \quad (i \neq j), \quad i, j = 1, 2, \cdots, N. \tag{3}$$

Throughout this paper, we make the following assumptions:

Assumption H$_1$**.** The nonlinear vector-valued functions $f(\cdot)$, $g(\cdot)$ and $h(\cdot)$ are continuous and satisfy

$$[f(\alpha) - f(\beta) - U_1(\alpha - \beta)]^T [f(\alpha) - f(\beta) - U_2(\alpha - \beta)] \leq 0, \quad \forall \alpha, \beta \in \mathbb{R}$$

$$[g(\alpha) - g(\beta) - V_1(\alpha - \beta)]^T [g(\alpha) - g(\beta) - V_2(\alpha - \beta)] \leq 0, \quad \forall \alpha, \beta \in \mathbb{R}$$

$$[h(\alpha) - h(\beta) - W_1(\alpha - \beta)]^T [h(\alpha) - h(\beta) - W_2(\alpha - \beta)] \leq 0, \quad \forall \alpha, \beta \in \mathbb{R}$$

where U_1, U_2, V_1, V_2, W_1 and W_2 are constant matrices.

Assumption H$_2$**.** The coupling matrix

$$G = \begin{bmatrix} N_{11} & N_{12} & \cdots & N_{1t} \\ N_{21} & N_{22} & \cdots & N_{2t} \\ \vdots & \vdots & \ddots & \vdots \\ N_{t1} & N_{t2} & \cdots & N_{tt} \end{bmatrix}$$

where $N_{ii} \in \mathbb{R}^{m_i \times m_i}$, $N_{ij} \in \mathbb{R}^{m_i \times m_j}$, $i, j = 1, 2, \cdots, t$, and assume that all rows in N_{ij} are the same, i.e., $N_{ij} = (u, u, \cdots, u)^T$, $u = (u_1, u_2, \cdots, u_{m_j})^T$ is a vector.

Definition 1. [5] The set $S = \{x = (x_1(s), x_2(s), \cdots, x_N(s)) \colon x_i(s) \in \mathcal{C}(\mathbb{N}[-d_M, 0], \mathbb{R}^n)$, $[x_1(s) = x_2(s) = \cdots = x_{m_1}(s)]$, $[x_{m_1+1}(s) = x_{m_1+2} = \cdots = x_{m_1+m_2}(s)]$, $\cdots$, $[x_{m_1+m_2+\cdots+m_{t-1}+1}(s) = x_{m_1+m_2+\cdots+m_{t-1}+2}(s) = \cdots = x_{m_1+m_2+\cdots+m_{t-1}+m_t}(s)]$, $m_1 + m_2 + \cdots + m_t = N\}$ is called the cluster synchronization manifold.

Definition 2. [5] A network with N nodes is said to realize cluster synchronization, if the N nodes split into several clusters, such as, $\{(1, 2, \cdots, m_1), (m_1 + 1, m_1 + 2, \cdots, m_1 + m_2), \cdots, (m_1 + m_2 + \cdots + m_{t-1} + 1, m_1 + m_2 + \cdots + m_{t-1} + 2, \cdots, m_1 + m_2 + \cdots + m_{t-1} + m_t), m_1 + m_2 + \cdots + m_t = N\}$ such that the nodes in the same cluster synchronize with one another, i.e., for the states $x_i(k)$ and $x_j(k)$ of arbitrary nodes i and j in the same cluster, $\lim_{k\to+\infty} \|x_i(k) - x_j(k)\| = 0$ holds.

Definition 3. [2] Let $\mathfrak{R}$ denote a ring, and define $T(\mathfrak{R}, K) = \{$the set of matrices with entries $\mathfrak{R}$ such that the sum of the entries in each row is equal to K for some $K \in \mathfrak{R}\}$.

Lemma 1. *[2] Let G be an $N \times N$ matrix in the set $T(\Re, K)$. When the $(N-1) \times (N-1)$ matrix H satisfies $MG = HM$, where $H = MGJ$, with*

$$M = \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & & \ddots & \\ & & & 1 & -1 \end{bmatrix}_{(N-1)\times N}, \quad J = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 1 \\ 0 & \cdots & 0 & 0 & 1 \\ 0 & \cdots & 0 & 0 & 0 \end{bmatrix}_{N\times(N-1)}$$

*in which **1** is the multiplicative identity of $\Re$. The matrix H can be rewritten explicitly as follows: $H_{(i,j)} = \sum_{k=1}^{j} G_{(i,k)} - G_{(i+1,k)}$, for $i, j \in 1, 2, \cdots, N-1$.*

Lemma 2. *[8] Under Assumption* H_2*, the $(N-t) \times (N-t)$ matrix $\widetilde{H}$ satisfies $\widetilde{M}G = \widetilde{H}\widetilde{M}$, where*

$$\widetilde{N} = \begin{bmatrix} N_{11} & & & \\ & N_{22} & & \\ & & \ddots & \\ & & & N_{tt} \end{bmatrix}_{N\times N}, \quad \widetilde{M} = \begin{bmatrix} M_1 & & & \\ & M_2 & & \\ & & \ddots & \\ & & & M_t \end{bmatrix}_{(N-t)\times N},$$

$$\widetilde{J} = \begin{bmatrix} J_1 & & & \\ & J_2 & & \\ & & \ddots & \\ & & & J_t \end{bmatrix}_{N\times(N-t)}$$

and $\widetilde{H} = \widetilde{M}\widetilde{N}\widetilde{J}$, $N_{ii} \in \mathbb{R}^{m_i \times m_i}$, $M_i \in \mathbb{R}^{(m_i-1)\times m_i}$, $J_i \in \mathbb{R}^{m_i \times (m_i-1)}$.

Lemma 3. *[14] Let $M \in \mathbb{R}^{n\times n}$ be a positive semidefinite matrix, $x_i \in \mathbb{R}^n$, and scalar constant $a_i \geq 0 (i = 1, 2, \cdots)$. If the series concerned is convergent, then the following inequality holds:*

$$\Big(\sum_{i=1}^{+\infty} a_i x_i\Big)^T M \Big(\sum_{i=1}^{+\infty} a_i x_i\Big) \leq \Big(\sum_{i=1}^{+\infty} a_i\Big) \sum_{i=1}^{+\infty} a_i x_i^T M x_i.$$

Lemma 4. *[12] Let $\otimes$ denote the notation of Kronecker product. Then, the following properties are satisfied in appropriate dimensions.*
(i) $(\alpha A) \otimes B = A \otimes (\alpha B)$
(ii) $(A + B) \otimes C = A \otimes C + B \otimes C$
(iii) $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$.

Let $x = (x_1, x_2, \cdots, x_N)^T$, where $x_i \in \mathbb{R}^n$, $i = 1, 2, \cdots, N$. Denote $\mathbf{M} = \widetilde{M} \otimes E_n$, where $\widetilde{M}$ is defined in Lemma 2. We define $d(x)$ as follows:

$$d(x) = \|\mathbf{M}x\|^2 = x^T \mathbf{M}^T \mathbf{M} x. \tag{4}$$

Lemma 5. *[8] $x \in S$ if and only if $\|\mathbf{M}x\| = 0$.*

3 Synchronization Criterion

In this section, we shall establish our main criterion based on the LMI approach. For presentation convenience, in the following, we denote

$$x_i(k) = (x_{i1}(k), x_{i2}(k), \cdots, x_{in}(k))^T \in \mathbb{R}^n, \quad i = 1, 2, \cdots, N.$$

$$x(k) = (x_1^T(k), x_2^T(k), \cdots, x_N^T(k))^T,$$

$$\mathbf{f}(x(k)) = (f^T(x_1(k)), f^T(x_2(k)), \cdots, f^T(x_N(k)))^T,$$

$$\mathbf{g}(x(k)) = (g^T(x_1(k)), g^T(x_2(k)), \cdots, g^T(x_N(k)))^T,$$

$$\mathbf{h}(x(k)) = (h^T(x_1(k)), h^T(x_2(k)), \cdots, h^T(x_N(k)))^T,$$

$$\tilde{\Sigma}_1 = (U_1^T U_2 + U_2^T U_1)/2,\ \tilde{\Sigma}_2 = (U_1^T + U_2^T)/2,\ \tilde{\Sigma}_3 = (V_1^T V_2 + V_2^T V_1)/2,$$

$$\tilde{\Sigma}_4 = (V_1^T + V_2^T)/2,\ \tilde{\Sigma}_5 = (W_1^T W_2 + W_2^T W_1)/2,\ \tilde{\Sigma}_6 = (W_1^T + W_2^T)/2,$$

$$\mathbf{G} = G \otimes \Gamma,\ \widetilde{H} = \widetilde{M}\widetilde{N}\widetilde{J},\ \mathbf{H} = \widetilde{H} \otimes \Gamma,\ \mathbf{\Sigma}_l = E_{N-t} \otimes \tilde{\Sigma}_l,\ l = 1, 2, 3, 4, 5, 6.$$

Using the Kronecker product, we can rewrite system (1) into a more compact form as

$$x(k+1) = \mathbf{f}(x(k)) + \mathbf{g}(x(k - d(k))) + \sum_{m=1}^{+\infty} \mu_m \mathbf{h}(x(k-m)) + \mathbf{G}x(k). \tag{5}$$

Theorem 1. *Under Assumption* H$_1$ *and* H$_2$, *the cluster synchronization manifold S of the discrete-time system (5) is globally attractive if there exist three positive definite matrices $\{P_l\}_{l=1}^3 \in \mathbb{R}^{(N-t)n\times(N-t)n}$ and three positive scalars $\{\delta_l\}_{l=1}^3$, such that the following LMI holds:*

$$\Pi = \begin{bmatrix} \Pi_{11} & \delta_1\mathbf{\Sigma}_2 & \delta_2\mathbf{\Sigma}_4 & 0 & \delta_3\mathbf{\Sigma}_6 & 0 & \mathbf{H}^T P_1 \\ * & -\delta_1 E_n & 0 & 0 & 0 & 0 & P_1 \\ * & * & \Pi_{33} & 0 & 0 & 0 & 0 \\ * & * & * & -P_2 & 0 & 0 & P_1 \\ * & * & * & * & -\delta_3 E_n + \tilde{\mu} P_3 & 0 & 0 \\ * & * & * & * & * & -\frac{1}{\tilde{\mu}} P_3 & P_1 \\ * & * & * & * & * & * & -P_1 \end{bmatrix} < 0, \tag{6}$$

where $\Pi_{11} = -P_1 - \delta_1\mathbf{\Sigma}_1 - \delta_2\mathbf{\Sigma}_3 - \delta_3\mathbf{\Sigma}_5$, $\Pi_{33} = -\delta_2 E_n + (d_M - d_m + 1)P_2$, and $\tilde{\mu} = \sum_{k=1}^{+\infty} \mu_k$.

Proof. We can get from Assumption H$_1$ that [13, 14]

$$\begin{bmatrix} \mathbf{M}x(k) \\ \mathbf{M}\mathbf{f}(x(k)) \end{bmatrix}^T \begin{bmatrix} \mathbf{\Sigma}_1 & -\mathbf{\Sigma}_2 \\ -\mathbf{\Sigma}_2 & E_n \end{bmatrix} \begin{bmatrix} \mathbf{M}x(k) \\ \mathbf{M}\mathbf{f}(x(k)) \end{bmatrix} \leq 0, \tag{7}$$

$$\begin{bmatrix} \mathbf{M}x(k) \\ \mathbf{M}\mathbf{g}(x(k)) \end{bmatrix}^T \begin{bmatrix} \mathbf{\Sigma}_3 & -\mathbf{\Sigma}_4 \\ -\mathbf{\Sigma}_4 & E_n \end{bmatrix} \begin{bmatrix} \mathbf{M}x(k) \\ \mathbf{M}\mathbf{g}(x(k)) \end{bmatrix} \leq 0, \tag{8}$$

and

$$\begin{bmatrix} \mathbf{M}x(k) \\ \mathbf{Mh}(x(k)) \end{bmatrix}^T \begin{bmatrix} \mathbf{\Sigma}_5 & -\mathbf{\Sigma}_6 \\ -\mathbf{\Sigma}_6 & E_n \end{bmatrix} \begin{bmatrix} \mathbf{M}x(k) \\ \mathbf{Mh}(x(k)) \end{bmatrix} \leq 0. \tag{9}$$

Now, we consider the following Lyapunov-Krasovskii functional candidate for system (5):

$$V(k) = \sum_{i=1}^{3} V_i(k), \tag{10}$$

where

$$V_1(k) = [\mathbf{M}x(k)]^T P_1 [\mathbf{M}x(k)],$$

$$V_2(k) = \sum_{i=-d_M+1}^{-d_m+1} \sum_{j=k-1+i}^{k-1} [\mathbf{Mg}(x(j))]^T P_2 [\mathbf{Mg}(x(j))],$$

$$V_3(k) = \sum_{i=1}^{+\infty} \mu_i \sum_{j=k-i}^{k-1} [\mathbf{Mh}(x(j))]^T P_3 [\mathbf{Mh}(x(j))].$$

Calculating the difference of $V_1(k)$ along the solutions of (5), we have

$$\begin{aligned} \mathit{\Delta} V_1(k) = \Big[& \mathbf{Mf}(x(k)) + \mathbf{Mg}(x(k-d(k))) + \sum_{m=1}^{+\infty} \mu_m \mathbf{Mh}(x(k-m)) \\ & + \mathbf{MG}x(k) \Big]^T P_1 \Big[\mathbf{Mf}(x(k)) + \mathbf{Mg}(x(k-d(k))) \\ & + \sum_{m=1}^{+\infty} \mu_m \mathbf{Mh}(x(k-m)) + \mathbf{MG}x(k) \Big] \\ & - [\mathbf{M}x(k)]^T P_1 [\mathbf{M}x(k)]. \end{aligned} \tag{11}$$

By the structure of $\mathbf{M}$, the following equalities are easy to verify:

$$\mathbf{MG} = (\widetilde{M} \otimes E_n)(G \otimes \mathit{\Gamma}) = \widetilde{M} G \otimes \mathit{\Gamma} = \widetilde{H} \widetilde{M} \otimes \mathit{\Gamma} = (\widetilde{H} \otimes \mathit{\Gamma})(\widetilde{M} \otimes E_n) = \mathbf{HM}.$$

So, we get

$$\begin{aligned} \mathit{\Delta} V_1(k) = \Big[& \mathbf{Mf}(x(k)) + \mathbf{Mg}(x(k-d(k))) + \sum_{m=1}^{+\infty} \mu_m \mathbf{Mh}(x(k-m)) \\ & + \mathbf{HM}x(k) \Big]^T P_1 \Big[\mathbf{Mf}(x(k)) + \mathbf{Mg}(x(k-d(k))) \\ & + \sum_{m=1}^{+\infty} \mu_m \mathbf{Mh}(x(k-m)) + \mathbf{HM}x(k) \Big] \\ & - [\mathbf{M}x(k)]^T P_1 [\mathbf{M}x(k)]. \end{aligned} \tag{12}$$

Calculating the difference of $V_2(k)$ and $V_3(k)$ along the solutions of (5), respectively, one has

$$\begin{aligned}\Delta V_2(k) = (d_M - d_m + 1)&[\mathbf{Mg}(x(k))]^T P_2[\mathbf{Mg}(x(k))]\\ &-[\mathbf{Mg}(x(k-d(k)))]^T P_2[\mathbf{Mg}(x(k-d(k)))]. \end{aligned} \tag{13}$$

$$\begin{aligned}\Delta V_3(k) = &\sum_{i=1}^{+\infty}\mu_i \sum_{j=k+1-i}^{k} [\mathbf{Mh}(x(j))]^T P_3[\mathbf{Mh}(x(j))]\\ &-\sum_{i=1}^{+\infty}\mu_i \sum_{j=k-i}^{k-1} [\mathbf{Mh}(x(j))]^T P_3[\mathbf{Mh}(x(j))]\\ =&\sum_{i=1}^{+\infty}\mu_i\Big([\mathbf{Mh}(x(k))]^T P_3[\mathbf{Mh}(x(k))]\\ &-[\mathbf{Mh}(x(k-i))]^T P_3[\mathbf{Mh}(x(k-i))]\Big)\\ =&\sum_{i=1}^{+\infty}\mu_i[\mathbf{Mh}(x(k))]^T P_3[\mathbf{Mh}(x(k))]\\ &-\sum_{i=1}^{+\infty}\mu_i[\mathbf{Mh}(x(k-i))]^T P_3[\mathbf{Mh}(x(k-i))]\\ \le\ &\tilde{\mu}[\mathbf{Mh}(x(k))]^T P_3[\mathbf{Mh}(x(k))]\\ &-\frac{1}{\tilde{\mu}}\Big[\sum_{m=1}^{+\infty}\mu_m\mathbf{Mh}(x(k-m))\Big]^T P_3\Big[\sum_{m=1}^{+\infty}\mu_m\mathbf{Mh}(x(k-m))\Big]. \end{aligned}\tag{14}$$

In deriving of $\Delta V_3(k)$, we have made use of Lemma 4.

Therefore, for three positive scalars $\{\delta_l\}_{l=1}^{3}$, it follows from (12)-(14) along with (7)-(9), we obtain

$$\Delta V(k) \le \zeta^T(k)\Pi\zeta(k) \tag{15}$$

where $\zeta(k) = ([\mathbf{M}x(k)]^T, [\mathbf{Mf}(x(k))]^T, [\mathbf{Mg}(x(k))]^T, [\mathbf{Mg}(x(k-d(k)))]^T, [\mathbf{Mh}(x(k))]^T, [\sum_{m=1}^{+\infty}\mu_m\mathbf{Mh}(x(k-m))]^T)^T$.

Noticing from (6) and (15) that $\Delta V(k) \le 0$ and $\Delta V(k) = 0$ if and only if $\zeta(k) \equiv 0$, which implies that $V(k) \le V(0)$, in other words, $V(k)$ is a bounded function. Thus, $\|\mathbf{M}x\| \to 0$, and the proof is completed. □

4 Numerical Example

In this section, an example will be illustrated the potential benefits and effectiveness of the developed designs on the cluster synchronization problems for discrete-time complex networks.

For simplicity, let us consider the system (1) of four nodes. Suppose that

$$n = 2,\ d(k) = 3 + (1 + (-1)^k)/2,\ \mu_m = 2^{-(m+3)},$$

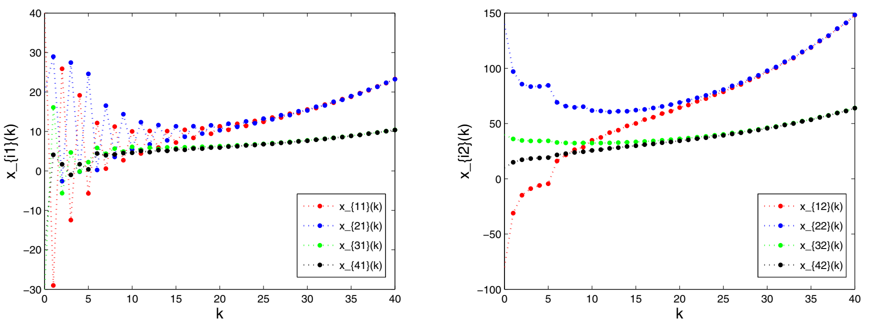


Fig. 1. Time responses of the state variables $x_{i1}(k)$ and $x_{i2}(k)$ $(1 \leq i \leq 4)$ in complex networks from initial point $\{(40, -80), (20, 140), (-30, 40), (-20, 10)\}$

$$\Gamma = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \; G = \begin{bmatrix} -0.5 & 0.498 & 0.001 & 0.001 \\ 0.498 & -0.5 & 0.001 & 0.001 \\ 0.002 & 0.002 & -0.4 & 0.396 \\ 0.002 & 0.002 & 0.396 & -0.4 \end{bmatrix}.$$

Let the nonlinear vector-valued functions be given by

$$f(x_i(k)) = (-0.4x_{i1}(k) + \tanh(0.5x_{i1}(k)) + 0.2x_{i2}(k), 0.8x_{i2}(k) - \tanh(0.6x_{i2}(k)))^T,$$

$$g(x_i(k)) = h(x_i(k)) = (0.2x_{i1}(k) - \tanh(0.1x_{i1}(k)), 0.3x_{i2}(k) - \tanh(0.2x_{i2}(k)))^T.$$

Then, it can be verified that $d_m = 3$, $d_M = 4$, $\tilde{\mu} = 1/8$, and

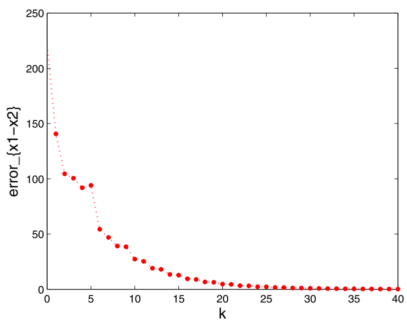
$$\tilde{\Sigma}_1 = \begin{bmatrix} -0.4 & 0.2 \\ 0 & 0.8 \end{bmatrix}, \tilde{\Sigma}_2 = \begin{bmatrix} 0.1 & 0.2 \\ 0 & 0.2 \end{bmatrix}, \tilde{\Sigma}_3 = \tilde{\Sigma}_5 = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.3 \end{bmatrix}, \tilde{\Sigma}_4 = \tilde{\Sigma}_6 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}.$$

By using the Matlab LMI Toolbox, (6) can be solved with the feasible solutions as follows:

$$P_1 = \begin{bmatrix} 0.2505 & 0.0005 & 0 & 0 \\ 0.0005 & 0.2192 & 0 & 0 \\ 0 & 0 & 0.2561 & -0.0002 \\ 0 & 0 & -0.0002 & 0.2206 \end{bmatrix}, P_2 = \begin{bmatrix} 0.7881 & 0.0002 & 0 & 0 \\ 0.0002 & 0.7886 & 0 & 0 \\ 0 & 0 & 0.7888 & 0.0001 \\ 0 & 0 & 0.0001 & 0.7888 \end{bmatrix},$$

$$P_3 = \begin{bmatrix} 0.1650 & -0.0003 & 0 & 0 \\ -0.0003 & 0.1588 & 0 & 0 \\ 0 & 0 & 0.1656 & -0.0003 \\ 0 & 0 & -0.0003 & 0.1590 \end{bmatrix},$$

$\delta_1 = 0.7896$, $\delta_2 = 2.3741$ and $\delta_3 = 0.9457$. Then, it follows from Theorem 1 that the system (1) with given parameters achieves cluster synchronization, which is further verified by the simulation result shown in Figs. 1, 2 and 3. Fig. 1 shows the cluster synchronization state, and the synchronization errors between the first



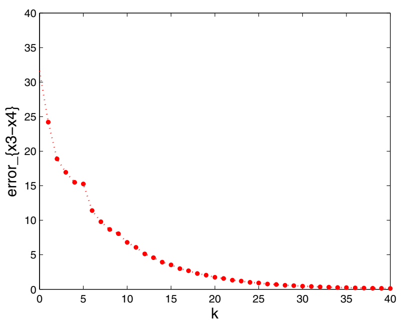


Fig. 2. Synchronous errors of the first two networks and synchronous errors of the last two networks

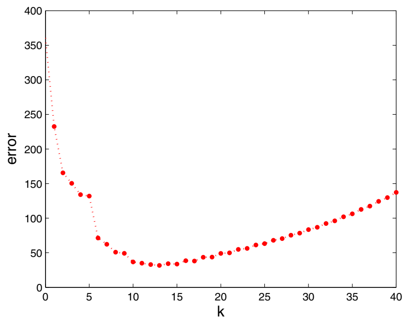


Fig. 3. Synchronous errors of the complex networks

two networks, the synchronization errors between the last two networks and the synchronization errors of the whole complex networks are illustrated by Fig. 2 and Fig. 3, respectively. The simulation results demonstrate the effectiveness of the developed approach for discrete-time complex networks.

5 Conclusion

The problem of cluster synchronization for discrete-time complex networks with mixed delays is investigate in this paper. The delay-interval-dependent criterion is established through utilizing a special coupling matrix and the Kronecker product. The obtained result is efficient which has been demonstrated by a numerical simulation example.

Acknowledgements

This work was partially supported by The National Natural Science Foundation of China under Grant No. 60974132 and the Postgraduate Innovation Foundation of Chongqing Jiaotong University.

References

1. Pecora, L., Carroll, T.: Synchronization in chaotic systems. Physical Review Letters 64, 821–824 (1990)
2. Wu, C., Chua, L.: Synchronization in an array of linearly coupled dynamical systems. IEEE Transactions on Circuits and Systems-I 42, 430–447 (1995)
3. Belykh, V., Belykh, I., Mosekilde, E.: Cluster synchronization modes in an ensemble of coupled chaotic oscillators. Physical Review E 63, 36216 (2001)
4. Qin, W., Chen, G.: Coupling schemes for cluster synchronization in coupled Josephson equations. Physica D 197, 375–391 (2004)
5. Ma, Z., Liu, Z., Zhang, G.: A new method to realize cluster synchronization in connected chaotic networks. Chaos 16, 023103 (2006)
6. Lu, X., Qin, B.: Adaptive cluster synchronization in complex dynamical networks. Physics Letters A 373, 3650–3658 (2009)
7. Wu, W., Zhou, W., Chen, T.: Cluster synchronization of linearly coupled complex networks under pinning control. IEEE Transactions on Circuits and Systems-I 56, 829–839 (2009)
8. Cao, J., Li, L.: Cluster synchronization in an array of hybrid coupled neural networks with delay. Neural Networks 22, 335–342 (2009)
9. Lu, W., Liu, B., Chen, T.: Cluster synchronization in networks of coupled nonidentical dynamical systems. Chaos 20, 013120 (2010)
10. Arenas, A., Guilera, A., Kurths, J., Morenob, Y., Zhou, C.: Synchronization in complex networks. Physics Reports 469, 93–153 (2008)
11. Song, Q., Liang, J., Wang, Z.: Passivity analysis of discrete-time stochastic neural networks with time-varying delays. Neurocomputing 72, 1782–1788 (2009)
12. Liang, J., Wang, Z., Liu, Y., Liu, X.: Robust synchronization of an array of coupled stochastic discrete-time delayed neural networks. IEEE Transactions on Neural Networks 19, 1910–1921 (2008)
13. Wang, Z., Wang, Y., Liu, Y.: Global synchronization for discrete-time stochastic complex networks with randomly occurred nonlinearities and mixed time delays. IEEE Transactions on Neural Networks 21, 11–25 (2010)
14. Liu, Y., Wang, Z., Liang, J., Liu, X.: Synchronization and state estimation for discrete-time complex networks with distributed delays. IEEE Transactions on System, Man, And Cybernetics-Part B 38, 1314–1325 (2008)

Stability Analysis of Fourth-Order Chua's Circuit

Chunfang Miao⋆ and Yunquan Ke

Department of Mathematics, Shaoxing University, shaoxing 312000, P.R. China
Tel.: +86-575-88345082
miaochf@usx.edu.cn

Abstract. In this paper, the global exponential stability of fourth-order Chua's circuit is investigated. Some sufficient conditions ensuring the existence and the global exponential stability of the equilibrium point are derived by selecting properly the system parameters and using eigenvalue, eigenvector and solution matrix property. Finally two illustrative examples are given to show the effectiveness of our results.

Keywords: Chua'circuit, stability, equilibrium point, eigenvector, solution matric.

1 Introduction

As a matter of fact, nonlinear electronic circuits play an important role in studying nonlinear phenomena such as chaotic dynamics and synchronization which have effective applications in secure communications. And Chua's circuit is considered to be the simplest autonomous nonlinear circuit generating chaotic signals and its characteristic diode is described by a continuous piecewise-linear function with three segments and two non differentiable break points. Thus, Chua's circuit has been widely used as the experimental vehicle to research on nonlinear science, and been studied in many applications such as chaos control and synchronization .However, the characteristics of nonlinear devices in practical circuit are always smooth and the implementation of piecewise-linear function requires a larger amount of circuitry compared with smooth cubic function. Therefore, it is significant to investigate Chua's circuit with a smooth cubic nonlinearity. Hartley proposed a smooth cubic nonlinearity to replace the piecewise-linear nonlinearity in Chua's circuit(see Ref.[1]). Since then, a lot of literatures and publications about Chua's circuit with a smooth cubic nonlinearity have appeared(see Refs.[2],[3],[4],[5],[6],[7],[8],[9]). Recently, the gallery of attractors and bifurcation diagrams of Chua's circuit with the piecewise-linear nonlinearity and with a smooth cubic nonlinearity were also presented in [10]. For the modified fourth-order Chua's circuit, it consists of two active elements, one linear negative conductance and one nonlinear resistor with an odd-symmetric piecewise-linear v-i characteristic. It was verified by experiment that hyper chaos

⋆ Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

phenomenon was exhibited (see Ref.[11]). Hyper chaos means the attractor of a dynamic system has more than one positive Lyapunov exponents, that is, its dynamics expands not only as a segment (one dimensional expansion) but also as a small area element (two-dimensional expansion), which increases the complexity, randomness and higher unpredictability. These properties make hyper chaotic systems have potential applications to secure communication. Lately the dynamics of fourth-order Chua's circuit containing one group of hyper chaotic attractor and one group of chaotic attractor, corresponding bifurcation diagrams and Lyapunov exponent spectra were presented by using practical experiments and simulations (see Ref.[12]). A new method of chaos control for the modified fourth-order Chua's circuit was given by using some results of dichotomous in nonlinear systems (see Ref.[13]).

In this paper, we will investigate the existence and the global exponential stability of the equilibrium point for fourth-order Chua's circuit by using eigenvalue, eigenvector and solution matrix property of coefficient matrix.

Consider the following Chua's circuit system(see Ref.[15])

$$\begin{cases} \frac{dx(t)}{dt} = \alpha[y(t) - x(t) - g(x(t))], \\ \frac{dy(t)}{dt} = x(t) - y(t) + z(t), \\ \frac{dz(t)}{dt} = -\beta[y(t) - w(t)], \\ \frac{dw(t)}{dt} = -\gamma_2[z(t) + \gamma_1 w(t)], \end{cases} \tag{1}$$

where $g(x) = m_1 x + \frac{1}{2}(m_0 - m_1)(|x + b_1| - |x - b_1|), \alpha, \beta, \gamma_1, \gamma_2, m_0, m_1, b_1$ are constants.

We give the initial values of system (1)

$$\begin{cases} x(t) = p(t), \\ y(t) = q(t), -\infty < t \le 0, \\ z(t) = r(t), \\ w(t) = s(t), \end{cases} \tag{2}$$

where $p(t), q(t), r(t)$ and $s(t)$ are bounded and continuous functions on $(-\infty, 0]$.

Let $X(t) = (x(t), y(t), z(t), w(t))^T, F(X(t)) = (-\alpha(m_0 - m_1)f(x(t)), 0, 0, 0)^T$,

$$\phi(t) = (p(t), q(t), r(t), s(t))^T, \qquad A = \begin{bmatrix} -\alpha b & \alpha & 0 & 0 \\ 1 & -1 & 1 & 0 \\ 0 & -\beta & 0 & \beta \\ 0 & 0 & -\gamma_2 & -\gamma_1\gamma_2 \end{bmatrix},$$

where $b = m_1 + 1, f(x) = \frac{1}{2}(|x + b_1| - |x - b_1|)$.

Rewrite the system (1) and (2) as

$$X'(t) = AX(t) + F(X(t)). \tag{3}$$

$$X(t) = \phi(t), \quad -\infty < t \le 0. \tag{4}$$

The paper is arranged as follows: In section 2, we will give some preliminaries. In section 3, the main results are presented. Finally, in section 4, we give two examples to illustrate our theory.

2 Definitions and Lemmas

In the following discussion, we always assume $\alpha > 0, \beta > 0, \gamma_1 > 0, \gamma_2 > 0$ and $b > 0$.

Definition 2.1. For vector function $V(t) = (v_1(t), v_2(t), \cdots, v_n(t))^T$ and $n \times n$ order matrix $G = (g_{ij})_{n\times n}$, we define norm as following, respectively

$$\|V(t)\| = (\sum_{i=1}^{n} |v_i(t)|^2)^{\frac{1}{2}}, \quad \|G\| = (\sum_{i,j=1}^{n} |g_{ij}|^2)^{\frac{1}{2}}.$$

Definition 2.2. Let $X^* = (x^*, y^*, z^*, w^*)^T \in R^4$. The point X^* is called an equilibrium point of system (1), if it satisfies the following equation

$$AX^* + F(X^*) = 0. \tag{5}$$

Definition 2.3. Let $X^* = (x^*, y^*, z^*, w^*)^T$ is an equilibrium point of system (1), $X(t)$ is a solution of system (1) with initial value (3). The equilibrium point X^* of system (1) is said to be global exponentially stable, if there exist constants $\lambda > 0$ and $\mu > 0$, such that

$$\|X(t) - X^*\| \le \mu \|\phi - X^*\| e^{-\lambda t}, t > 0,$$

where $\|\phi - X^*\| = \sup\limits_{-\infty<t\le 0} \{[(p(t)-x^*)^2 + (q(t)-x^*)^2 + (r(t)-x^*)^2 + (s(t)-x^*)^2]^{\frac{1}{2}}\}$.

Lemma 2.1. For function $f(x) = \frac{1}{2}(|x + b_1| - |x - b_1|)$, we have

$$|f(x_1) - f(x_2)| \le |x_1 - x_2|, \quad \forall x_1, x_2 \in R.$$

Lemma 2.2. For the system (1), if $\frac{\alpha^2 b}{\beta\gamma_2} = \frac{\alpha}{\gamma_1\gamma_2 - \alpha b} = \gamma_1\gamma_2 - 1$, then the eigenvalue and corresponding eigenvector of matrix A, respectively

$\lambda_1 = -\alpha b, \quad \lambda_2 = -\gamma_1\gamma_2, \quad \lambda_3 = -\frac{1}{2}(1 - \sqrt{1 - 4(\beta + \beta\gamma_2 - \alpha)}), \quad \lambda_4 = -\frac{1}{2}(1 + \sqrt{1 - 4(\beta + \beta\gamma_2 - \alpha)})$,

$V_1 = (1, 0, -1, \frac{\alpha b}{\beta})^T, \quad V_2 = (1 - \gamma_1\gamma_2, 1, 0, 1)^T$,

$V_3 = (\alpha, \lambda_3 + \alpha b, \alpha b(1 + \lambda_3) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_3)^T$,

$V_4 = (\alpha, \lambda_4 + \alpha b, \alpha b(1 + \lambda_4) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_4)^T$.

Proof. By calculation, we have

$$|\lambda E - A| = (\lambda + \alpha b)(\lambda + \gamma_1\gamma_2)(\lambda^2 + \lambda + \beta + \beta\gamma_2 - \alpha) + (\lambda + \alpha b)[\beta\gamma_2(1 - \gamma_1\gamma_2) + \alpha^2 b] - \alpha b(\alpha^2 b - \alpha\gamma_1\gamma_2) - \alpha\beta\gamma_2.$$

If $\frac{\alpha^2 b}{\beta\gamma_2} = \frac{\alpha}{\gamma_1\gamma_2 - \alpha b} = \gamma_1\gamma_2 - 1$, then characteristic equation of A is

$$|\lambda E - A| = (\lambda + \alpha b)(\lambda + \gamma_1\gamma_2)(\lambda^2 + \lambda + \beta + \beta\gamma_2 - \alpha) = 0. \tag{6}$$

We can obtain the eigenvalue of A,

$\lambda_1 = -\alpha b, \lambda_2 = -\gamma_1\gamma_2, \lambda_3 = -\frac{1}{2}(1 - \sqrt{1 - 4(\beta + \beta\gamma_2 - \alpha)}), \lambda_4 = -\frac{1}{2}(1 + \sqrt{1 - 4(\beta + \beta\gamma_2 - \alpha)})$.

For the eigenvalue $\lambda_1 = -\alpha b$, From

$$(\lambda_1 E - A)U = \begin{bmatrix} 0 & -\alpha & 0 & 0 \\ -1 & 1 - \alpha b & -1 & 0 \\ 0 & \beta & -\alpha b & -\beta \\ 0 & 0 & \gamma_2 & \gamma_1\gamma_2 - \alpha b \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = 0. \tag{7}$$

We can obtain corresponding eigenvector $V_1 = (1, 0, -1, \frac{\alpha b}{\beta})^T$.

Similarly, we can obtain the eigenvalue $\lambda_2, \lambda_3, \lambda_4$ corresponding the eigenvector are V_2, V_3, V_4, respectively. $\square$

Lemma 2.3. For the system (1), if $\frac{\alpha^2 b}{\beta\gamma_2} = \frac{\alpha}{\gamma_1\gamma_2 - \alpha b} = \gamma_1\gamma_2 - 1$, $0 < 4(\beta + \beta\gamma_2 - \alpha) \leq 1$, then $\|exp(At)\| \leq Me^{-\sigma t}, t \geq 0$.
where $\sigma = \min\{\alpha b, \gamma_1\gamma_2, \frac{1}{2}(1 - \sqrt{1 - 4(\beta + \beta\gamma_2 - \alpha)})\}, M = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3\|^2 + \|V_4\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3, V_4]^{-1}\|$.($V_1, V_2, V_3, V_4$ are given by Lemma 2.2).

Proof. For linear differential equations

$$X'(t) = AX(t). \tag{8}$$

From Lemma 2.2, we can obtain the eigenvalue and eigenvector of A $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ and V_1, V_2, V_3, V_4, respectively.

If $0 < 4(\beta + \beta\gamma_2 - \alpha) \leq 1$, then λ_3, λ_4 are real roots and $\lambda_3 < 0, \lambda_4 < 0$, we can obtain its fundamental solution matrix of equation $\phi(t) = [e^{\lambda_1 t}V_1, e^{\lambda_2 t}V_2, e^{\lambda_3 t}V_3, e^{\lambda_4 t}V_4]$.

Since $exp(At) = \phi(t)\phi^{-1}(0)$, we can obtain

$$\|exp(At)\| = \|\phi(t)\phi^{-1}(0)\| \leq \|\phi(t)\| \cdot \|\phi^{-1}(0)\| \leq Me^{-\sigma t}.$$

where $\sigma = \min\{\alpha b, \gamma_1\gamma_2, \frac{1}{2}(1 - \sqrt{1 - 4(\beta + \beta\gamma_2 - \alpha)})\}$,

$$M = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3\|^2 + \|V_4\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3, V_4]^{-1}\|. \qquad \square$$

With the same proof methods of Lemmas 2.2 and Lemmas 2.3,we can obtain the following Lemmas.Here we omit the proofs.

Lemma 2.3*. For the system (1), if $\frac{\alpha^2 b}{\beta\gamma_2} = \frac{\alpha}{\gamma_1\gamma_2 - \alpha b} = \gamma_1\gamma_2 - 1$, $1 - 4(\beta + \beta\gamma_2 - \alpha) < 0$, then $\|exp(At)\| \leq M^* e^{-\sigma^* t}, t \geq 0$,
where $\sigma^* = \min\{\alpha b, \gamma_1\gamma_2, \frac{1}{2}\}$,$v = \frac{1}{2}\sqrt{4(\beta + \beta\gamma_2 - \alpha) - 1}$,
$M^* = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3^*\|^2 + \|V_4^*\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3^*, V_4^*]^{-1}\|$,
$V_3^* = (\alpha cos(vt), (\alpha b - \frac{1}{2})cos(vt) - vsin(vt), \frac{1}{2}(\alpha b - \beta - \beta\gamma_2)cos(vt) - v\alpha bsin(vt), \frac{1}{2\beta}$
$(2\alpha b(\alpha - \beta\gamma_2) + \beta\gamma_2)cos(vt) + v\gamma_2 sin(vt))^T$,
$\overrightarrow{V}_4^* = (\alpha sin(vt), (\alpha b - \frac{1}{2})sin(vt) + vcos(vt), \frac{1}{2}(\alpha b - \beta - \beta\gamma_2)sin(vt) + v\alpha bcos(vt), \frac{1}{2\beta}$
$(2\alpha b(\alpha - \beta\gamma_2) + \beta\gamma_2)sin(vt) - v\gamma_2 cos(vt))^T$,
(V_1, V_2,are given by Lemma 2.2).

Lemma 2.4. For the system (1), if $\frac{\beta\gamma_2}{b\alpha} = \frac{\alpha + \beta\gamma_1\gamma_2 - \beta}{\alpha} = \gamma_1\gamma_2 - b\alpha$, $0 < 4(\beta + \beta\gamma_2 - \alpha) \leq (\gamma_1\gamma_2)^2$, then
1) the eigenvalue and corresponding eigenvector of A, respectively
$\lambda_1 = -\alpha b, \quad \lambda_2 = -1, \quad \lambda_3 = -\frac{1}{2}(\gamma_1\gamma_2 - \sqrt{(\gamma_1\gamma_2)^2 - 4(\beta + \beta\gamma_2 - \alpha)})$,
$\lambda_4 = -\frac{1}{2}(\gamma_1\gamma_2 + \sqrt{(\gamma_1\gamma_2)^2 - 4(\beta + \beta\gamma_2 - \alpha)})$,
$V_1 = (1, 0, -1, \frac{\alpha b}{\beta})^T, \quad V_2 = (\alpha, b\alpha - 1, -\alpha, \frac{\alpha}{\beta} + b\alpha - 1)^T$,
$V_3 = (\alpha, \lambda_3 + \alpha b, \alpha b(1 + \lambda_3) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_3)^T$,
$V_4 = (\alpha, \lambda_4 + \alpha b, \alpha b(1 + \lambda_4) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_4)^T$.
2) $\|exp(At)\| \leq M_1 e^{-\sigma_1 t}, t \geq 0$,

where $\sigma_1 = \min\{\alpha b, 1, \frac{1}{2}(\gamma_1\gamma_2 - \sqrt{(\gamma_1\gamma_2)^2 - 4(\beta + \beta\gamma_2 - \alpha)})\}$,
$M_1 = \{\|wV_1\|^2 + \|V_2\|^2 + \|V_3\|^2 + \|V_4\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3, V_4]^{-1}\|$.

Lemma 2.4*. For the system (1), if $\frac{\beta\gamma_2}{b\alpha} = \frac{\alpha+\beta\gamma_1\gamma_2-\beta}{\alpha} = \gamma_1\gamma_2 - b\alpha$, $(\gamma_1\gamma_2)^2 - 4(\beta + \beta\gamma_2 - \alpha) < 0$, then
1) the eigenvalue and corresponding eigenvector of A, respectively
$\lambda_1 = -\alpha b, \lambda_2 = -1, \lambda_3 = -\frac{1}{2}(\gamma_1\gamma_2 - i\sqrt{4(\beta + \beta\gamma_2 - \alpha) - (\gamma_1\gamma_2)^2})$,
$\lambda_4 = -\frac{1}{2}(\gamma_1\gamma_2 + i\sqrt{4(\beta + \beta\gamma_2 - \alpha) - (\gamma_1\gamma_2)^2})$,
$V_1 = (1, 0, -1, \frac{\alpha b}{\beta})^T, V_2 = (\alpha, b\alpha - 1, -\alpha, \frac{\alpha}{\beta} + b\alpha - 1)^T$,
$V_3 = (\alpha, \lambda_3 + \alpha b, \alpha b(1 + \lambda_3) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_3)^T$,
$V_4 = (\alpha, \lambda_4 + \alpha b, \alpha b(1 + \lambda_4) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_4)^T$.
2) $\|exp(At)\| \le M_1^* e^{-\sigma_1^* t}, t \ge 0$,
where $\sigma_1^* = \min\{\alpha b, 1, \frac{1}{2}\gamma_1\gamma_2\}, v = \frac{1}{2}\sqrt{4(\beta + \beta\gamma_2 - \alpha) - (\gamma_1\gamma_2)^2}$.
$M_1^* = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3^*\|^2 + \|V_4^*\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3^*, V_4^*]^{-1}\|$,
$V_3^* = (\alpha cos(vt), (\alpha b - \frac{1}{2})cos(vt) - vsin(vt), \frac{1}{2}(\alpha b - \beta - \beta\gamma_2)cos(vt) - v\alpha bsin(vt), \frac{1}{2\beta}$
$(2\alpha b(\alpha - \beta\gamma_2) + \beta\gamma_2)cos(vt) + v\gamma_2 sin(vt))^T$,
$V_4^* = (\alpha sin(vt), (\alpha b - \frac{1}{2})sin(vt) + vcos(vt), \frac{1}{2}(\alpha b - \beta - \beta\gamma_2)sin(vt) + v\alpha bcos(vt), \frac{1}{2\beta}$
$(2\alpha b(\alpha - \beta\gamma_2) + \beta\gamma_2)sin(vt) - v\gamma_2 cos(vt))^T$

Lemma 2.5. For the system (1), if $\frac{\alpha}{1-\gamma_1\gamma_2} = \frac{\beta - b\alpha\beta - \alpha}{\beta\gamma_2} = b\alpha - \gamma_1\gamma_2$, $0 < 4(\beta + \beta\gamma_2 - \alpha) \le (b\alpha)^2$, then
1) the eigenvalue and corresponding eigenvector of A, respectively
$\lambda_1 = -1, \lambda_2 = -\gamma_1\gamma_2, \lambda_3 = -\frac{1}{2}(b\alpha - \sqrt{(b\alpha)^2 - 4(\beta + \beta\gamma_2 - \alpha)})$,
$\lambda_4 = -\frac{1}{2}(b\alpha + \sqrt{(b\alpha)^2 - 4(\beta + \beta\gamma_2 - \alpha)})$,
$V_1 = (\alpha, b\alpha - 1, -\alpha, b\alpha + \frac{\alpha}{\beta} - 1)^T, \quad V_2 = (1 - \gamma_1\gamma_2, 1, 0, 1)^T$,
$V_3 = (\alpha, \lambda_3 + \alpha b, \alpha b(1 + \lambda_3) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_3)^T$,
$V_4 = (\alpha, \lambda_4 + \alpha b, \alpha b(1 + \lambda_4) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_4)^T$.
2) $\|exp(At)\| \le M_2 e^{-\sigma_2 t}, t \ge 0$,
where $\sigma_2 = \min\{1, \gamma_1\gamma_2, \frac{1}{2}(\alpha b - \sqrt{(\alpha b)^2 - 4(\beta + \beta\gamma_2 - \alpha)})\}$,
$M_2 = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3\|^2 + \|V_4\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3, V_4]^{-1}\|$.

Lemma2.5*. For the system (1), if $\frac{\alpha}{1-\gamma_1\gamma_2} = \frac{\beta - b\alpha\beta - \alpha}{\beta\gamma_2} = b\alpha - \gamma_1\gamma_2$, $(b\alpha)^2 - 4(\beta + \beta\gamma_2 - \alpha) < 0$, then
1) the eigenvalue and corresponding eigenvector of A, respectively
$\lambda_1 = -1, \lambda_2 = -\gamma_1\gamma_2, \lambda_3 = -\frac{1}{2}(b\alpha - i\sqrt{4(\beta + \beta\gamma_2 - \alpha) - (b\alpha)^2})$,
$\lambda_4 = -\frac{1}{2}(b\alpha + i\sqrt{4(\beta + \beta\gamma_2 - \alpha) - (b\alpha)^2})$,
$V_1 = (\alpha, b\alpha - 1, -\alpha, b\alpha + \frac{\alpha}{\beta} - 1)^T, V_2 = (1 - \gamma_1\gamma_2, 1, 0, 1)^T$,
$V_3 = (\alpha, \lambda_3 + \alpha b, \alpha b(1 + \lambda_3) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_3)^T$,
$V_4 = (\alpha, \lambda_4 + \alpha b, \alpha b(1 + \lambda_4) - \beta(1 + \gamma_2), \frac{\alpha b}{\beta}(\alpha - \beta\gamma_2) - \gamma_2\lambda_4)^T$.
2) $\|exp(At)\| \le M_2^* e^{-\sigma_2^* t}, t \ge 0$,
where $\sigma_1^* = \min\{1, \gamma_1\gamma_2, \frac{1}{2}\alpha b\}, v = \frac{1}{2}\sqrt{4(\beta + \beta\gamma_2 - \alpha) - (\alpha b)^2}$.
$M_2^* = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3^*\|^2 + \|V_4^*\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3^*, V_4^*]^{-1}\|$,

$V_3^* = (\alpha cos(vt), (\alpha b - \frac{1}{2})cos(vt) - vsin(vt), \frac{1}{2}(\alpha b - \beta - \beta\gamma_2)cos(vt) - v\alpha bsin(vt), \frac{1}{2\beta}$
$(2\alpha b(\alpha - \beta\gamma_2) + \beta\gamma_2)cos(vt) + v\gamma_2 sin(vt))^T,$
$V_4^* = (\alpha sin(vt), (\alpha b - \frac{1}{2})sin(vt) + vcos(vt), \frac{1}{2}(\alpha b - \beta - \beta\gamma_2)sin(vt) + v\alpha bcos(vt), \frac{1}{2\beta}$
$(2\alpha b(\alpha - \beta\gamma_2) + \beta\gamma_2)sin(vt) - v\gamma_2 cos(vt))^T,$

3 Main Results

Theorem 3.1. For the system (1),

1) If $k = \frac{(m_1 - m_0)(1+\gamma_1)}{m_1 + \gamma_1 + m_1\gamma_1} < 1$, then the system (1) has a unique equilibrium point, and it is $(0,0,0,0)^T$.
2) If $k = \frac{(m_1 - m_0)(1+\gamma_1)}{m_1 + \gamma_1 + m_1\gamma_1} > 1$, then the system (1) has three equilibrium points, and they are

$$(0,0,0,0)^T, (kb_1, \frac{kb_1}{1+\gamma_1}, \frac{kb_1}{1+\gamma_1}, \frac{-kb_1\gamma_1}{1+\gamma_1})^T, (-kb_1, \frac{-kb_1}{1+\gamma_1}, \frac{-kb_1}{1+\gamma_1}, \frac{kb_1\gamma_1}{1+\gamma_1})^T.$$

3) If $k = \frac{(m_1 - m_0)(1+\gamma_1)}{m_1 + \gamma_1 + m_1\gamma_1} = 1$, then the system (1) has infinite number of equilibrium points.

Proof. From definition 2.2, we know that the equilibrium point $X = (x, y, z, w)^T$ of system (1) satisfies the following equation

$$AX + F(X) = 0, or \begin{cases} -(m_1 + 1)x + y - (m_0 - m_1)f(x) = 0, \\ x - y + z = 0, \\ y - w = 0, \\ z + \gamma_1 w = 0. \end{cases} \tag{9}$$

From (9), we can obtain

$$(x, y, z, w)^T = (kf(x), \frac{x}{1+\gamma_1}, \frac{x}{1+\gamma_1}, \frac{-\gamma_1 x}{1+\gamma_1})^T \tag{10}$$

where $k = \frac{(m_1 - m_0)(1+\gamma_1)}{m_1 + \gamma_1 + m_1\gamma_1}$. Since $f(x) = \frac{1}{2}(|x + b_1| - |x - b_1|)$, from (10), by calculation, we have

1) If $k < 1$, then the system (1) has a unique equilibrium point $(0,0,0,0)^T$.
2) If $k > 1$, the system (1) has three equilibrium points, i.e.,
$(0,0,0,0)^T, (kb_1, \frac{kb_1}{1+\gamma_1}, \frac{kb_1}{1+\gamma_1}, \frac{-kb_1\gamma_1}{1+\gamma_1})^T, (-kb_1, \frac{-kb_1}{1+\gamma_1}, \frac{-kb_1}{1+\gamma_1}, \frac{kb_1\gamma_1}{1+\gamma_1})^T.$
3) If $k = \frac{(m_1 - m_0)(1+\gamma_1)}{m_1 + \gamma_1 + m_1\gamma_1} = 1$, then the system (1) has infinite number of equilibrium points. □

Theorem 3.2. For the system (1), under the conditions of the Lemma 2.3, if $\sigma - M|\alpha(m_0 - m_1)| > 0$, then the equilibrium point of system (1) is globally exponentially stably, where M, σ is given of Lemma 2.3.

Proof. From Theorem 3.1, system (1) exists equilibrium point $X^* = (x^*, y^*, z^*, w^*)^T$. Let

$$U(t) = \begin{cases} X(t) - X^*, t > 0, \\ \phi(t) - X^*, -\infty < t \le 0, \end{cases} V(U(t)) = F(X(t)) - F(X^*).$$

From (3) and (5), we have

$$U'(t) = AU(t) + V(U(t)). \tag{11}$$

$$U(t) = e^{At}U(0) + \int_0^t e^{A(t-s)}V(U(s))ds. \tag{12}$$

By Lemma 2.3, from (12) we have

$$\begin{aligned}\|U(t)\| &\le M\|U(0)\|e^{-\sigma t} + M\int_0^t e^{-\sigma(t-s)} \cdot \|V(U(s))\|ds \\ &\le M\|U(0)\|e^{-\sigma t} + M|\alpha(m_0 - m_1)|\int_0^t e^{-\sigma(t-s)}\|U(s)\|ds. \end{aligned} \tag{13}$$

Let $P(t) = M\|U(0)\|e^{-\sigma t} + M|\alpha(m_0 - m_1)|\int_0^t e^{-\sigma(t-s)}\|U(s)\|ds, t \ge 0.$
We have

$$\frac{dP(t)}{dt} = -\sigma P(t) + M|\alpha(m_0 - m_1)| \cdot \|U(t)\| \le (-\sigma + M|\alpha(m_0 - m_1)|)P(t).$$

Then

$$P(t) \le M\|U(0)\|e^{-(\sigma - M|\alpha(m_0-m_1)|)t}, t > 0.$$

and

$$\|U(t)\| \le M\|U(0)\|e^{-(\sigma - M|\alpha(m_0-m_1)|)t}, t > 0,$$

i.e.,

$$\|X - X^*\| \le M\|\phi - X^*\|e^{-(\sigma - M|\alpha(m_0-m_1)|)t}, t > 0. \tag{14}$$

Since $\sigma - M|\alpha(m_0 - m_1)| > 0, M > 0$, from definition 2.3, we know that the equilibrium point X^* of system (1) is globally exponentially stable. □

By Lemma 2.3*, Lemma 2.4, Lemma 2.4*, Lemma 2.5 and Lemma 2.5*, we may obtain the following Theorem 3.3 - Theorem 3.7 with the same proof methods of Theorem 3.2. Here we omit the proof.

Theorem 3.3. For the system (1), under the conditions of the Lemma 2.3*, if $\sigma^* - M^*|\alpha(m_0 - m_1)| > 0$, then the equilibrium point of system (1) is globally exponentially stably.

Theorem 3.4. For the system (1), under the conditions of the Lemma 2.4, if $\sigma_1 - M_1|\alpha(m_0 - m_1)| > 0$, then the equilibrium point of system (1) is globally exponentially stably.

Theorem 3.5. For the system (1), under the conditions of the Lemma 2.4*, if $\sigma_1^* - M_1^*|\alpha(m_0 - m_1)| > 0$, then the equilibrium point of system (1) is globally exponentially stably.

Theorem 3.6. For the system (1), under the conditions of the Lemma 2.5, if $\sigma_2 - M_2|\alpha(m_0 - m_1)| > 0$, then the equilibrium point of system (1) is globally exponentially stably.

Theorem 3.7. For the system (1), under the conditions of the Lemma 2.5*, if $\sigma_2^* - M_2^*|\alpha(m_0 - m_1)| > 0$, then the equilibrium point of system (1) is globally exponentially stably.

Remark 1. Theorem 3.2 − 3.7 are developed under different assumptions. They provide different sufficient conditions ensuring the equilibrium point of system (1) to be globally exponentially stable. Therefore, we can select suitable theorems for a fourth-order Chuas circuit system to determine its globally exponential stability.

4 Example

Example 4.1. Consider the following fourth-order Chuas circuit system:

$$\begin{cases} \frac{dx(t)}{dt} = \alpha[y(t) - x(t) - g(x(t))], \\ \frac{dy(t)}{dt} = x(t) - y(t) + z(t), \\ \frac{dz(t)}{dt} = -\beta[y(t) - w(t)], \qquad t \geq t_0, \\ \frac{dw(t)}{dt} = -\gamma_2[z(t) + \gamma_1 w(t)], \end{cases} \tag{15}$$

$$g(x) = m_1 x + \frac{1}{2}(m_0 - m_1)(|x + b_1| - |x - b_1|), \tag{16}$$

where $\gamma_1 = 1, \gamma_2 = 2, \alpha = \frac{1}{12}, \beta = \frac{23}{288}, m_0 = \frac{177}{8}, m_1 = 22$.
We can obtain $b = 23$, $\frac{\alpha^2 b}{\beta\gamma_2} = \frac{\alpha}{\gamma_1\gamma_2 - \alpha b} = \gamma_1\gamma_2 - 1 = 1, 0 < 4(\beta + \beta\gamma_2 - \alpha) = \frac{3}{8} < 1$.

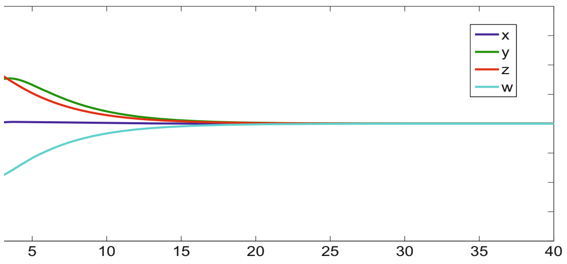


Fig. 1. The state diagram of system (15)

From Lemma 2.2, Lemma 2.3, we have

$\lambda_1 = -\frac{23}{12}, \quad \lambda_2 = -2, \quad \lambda_3 = -\frac{1}{2}(1 - \frac{\sqrt{6}}{4}), \quad \lambda_4 = -\frac{1}{2}(1 + \frac{\sqrt{6}}{4})$,
$V_1 = (1, 0, -1, 24)^T, V_2 = (-1, 1, 0, 1)^T, V_3 = (\frac{1}{12}, \frac{17}{12} + \frac{\sqrt{6}}{8}, \frac{207}{288} + \frac{23\sqrt{6}}{96}, \frac{1}{23} - \frac{\sqrt{6}}{2})^T$,
$V_4 = (\frac{1}{12}, \frac{17}{12} - \frac{\sqrt{6}}{8}, \frac{207}{288} - \frac{23\sqrt{6}}{96}, \frac{1}{23} + \frac{\sqrt{6}}{2})^T$, $\sigma = \min\{\frac{23}{12}, 2, \frac{1}{2}(1 - \frac{\sqrt{6}}{4})\} = \frac{1}{2}(1 - \frac{\sqrt{6}}{4})$,
$M = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3\|^2 + \|V_4\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3, V_4]^{-1}\| < 3\sqrt{21}$.

Obviously, we have

$$\sigma - M|\alpha(m_0 - m_1)| > \frac{16 - 4\sqrt{6} - \sqrt{21}}{32} > 0, \quad \frac{(m_1 - m_0)(1 + \gamma_1)}{m_1 + \gamma_1 + m_1\gamma_1} = -\frac{1}{180} < 1,$$

From (15), we can get the equation of the equilibriums

$$\begin{cases} -23x + y - \frac{1}{8}f(x) = 0, \\ x - y + z = 0, \\ y - w = 0, \\ z + w = 0. \end{cases} \tag{17}$$

From (17), by calculation, we can solve the unique equilibrium point $(0, 0, 0, 0)^T$. Figure 1 shows the state diagram of system (15) with the initial values is $(-2, -1, 1, 2)$, i.e., the equilibrium point is globally exponentially stable. Evidently, this consequence is coincident with the results of Theorem 3.1 and Theorem 3.2.

Example 4.2. For system (15), let $\gamma_1 = 2, \gamma_2 = 3, \alpha = \frac{40}{9}, \beta = \frac{8}{3}, m_0 = -\frac{11.01}{20}, m_1 = -\frac{11}{20}$. We can obtain $b = \frac{9}{20}$, and

$$\frac{\beta\gamma_2}{b\alpha} = \frac{\alpha + \beta\gamma_1\gamma_2 - \beta}{\alpha} = \gamma_1\gamma_2 - b\alpha = 4,\ 0 < 4(\beta + \beta\gamma_2 - \alpha) = \frac{224}{9} < (\gamma_1\gamma_2)^2 = 36.$$

From Lemma 2.4, we have
$\lambda_1 = -2, \quad \lambda_2 = -1, \quad \lambda_3 = -\frac{4}{3}, \quad \lambda_4 = -\frac{14}{3}, \quad \sigma = \min\{2, 1, \frac{4}{3}\} = 1,$
$M_1 = \{\|V_1\|^2 + \|V_2\|^2 + \|V_3\|^2 + \|V_4\|^2\}^{1/2} \cdot \|[V_1, V_2, V_3, V_4]^{-1}\| < 221.$
Obviously, we have
$\sigma_1 - M_1|\alpha(m_0 - m_1)| > \frac{79}{450} > 0, \quad \frac{(m_1 - m_0)(1+\gamma_1)}{m_1 + \gamma_1 + m_1\gamma_1} = \frac{3}{700} < 1,$
From (15), we can get the equation of the equilibriums

$$\begin{cases} -9x + 20y + 0.01f(x) = 0, \\ x - y + z = 0, \\ y - w = 0, \\ z + 2w = 0. \end{cases} \tag{18}$$

From (18), by calculation, we can solve the unique equilibrium point $(0, 0, 0, 0)^T$. Figure 2 shows the equilibrium point of the state diagram of system (15) with the initial values which is $(-2, -1, 1.5, 1)$ is globally exponentially stable. Evidently, this consequence is coincident with the results of Theorem 3.1 and Theorem 3.4.

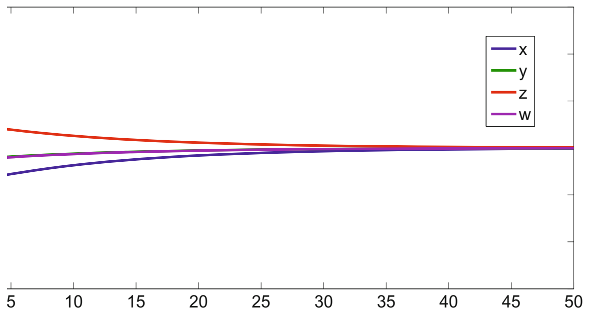


Fig. 2. The state diagram of system of Example 4.2

5 Conclusions

Under different assumption conditions, six theorems are given to ensure the existence, uniqueness and the global exponential stability of the equilibrium point for a fourth-order Chuas circuit system by selecting properly the system parameters and using eigenvalue, eigenvector and solution matrix property, and which algebra conditions are easily verifiable.

Acknowledgements

This work is supported by the Natural Science Foundation of Zhejiang Province (No. Y6100096),the education office of Zhejiang Province(Y200805543),and the National Natural Science Foundation of China(No.10871226, No.10875078).

References

1. Hartley, T.T.: The duffing double scroll. In: Proc. American Control Conf., vol. 1, pp. 419–423 (1989)
2. Algaba, A., Freire, E., Gamero, E., Rodriguez Luis, A.J.: On the Takens- Bogdanov bifurcation in the Chua's equation. IEICE Trans. Fund. Electron. Commun. Comput. Sci. E82-A, 1722–1728 (1999)
3. Algaba, A., Maestre, M., Gamero, E.: On the Hopf-pitchfork bifurcation in the Chua's equation. Int. J. Bifurcation and Chaos 10, 291–305 (2000)
4. Algaba, A., Garcia, C., Maestre, M.: Cusps in a strong resonances organized for a degenerate TakensC Bogdanov in the Chua's equations. IEICE Trans. Fund. Electron. Commun. Comput. Sci. E84-A, 2138–2144 (2001)
5. Ge, S.S., Wang, C.: Adaptive control of uncertain Chua's circuits. IEEE Trans. Circuits Syst. IFund. Th. Appl. 47, 1397–1402 (2000)
6. Jiang, M.J., Chen, C.L., Chen, C.K.: Sliding mode control of chaos in the cubic Chua's circuit system. Int. J. Bifurcation and Chaos 12, 1437–1449 (2002)
7. Moez, F.: Model-independent adaptive control of Chua's system with cubic nonlinearity. Int. J. Bifurcation and Chaos 14, 4249–4263 (2004)
8. Eltawil, A.M., Elwakil, A.S.: lowvoltage chaotic oscillator with an approximate cubic nonlinearity. AEU-Int. J. Electron 53, 158–160 (1999)
9. O'Donoghue, K., Forbes, P., Kennedy, M.P.: A fast and simple implementation of Chua's oscillator with cubic-like nonlinearity. Int. J. Bifurcation and Chaos 15, 2959–2972 (2005)
10. Tsuneda, A.: A gallery of attractors from smooth Chua's equation. Int. J. Bifurcation and Chaos 15, 1–49 (2005)
11. Matsumoto, T., Chua, L.O., Kobayashi, K.: Hyperchaos: Laboratory experiment and numerical confirmation. IEEE Trans. Circuits Syst. 33, 1143–1147 (1986)
12. Thamilmaran, K., Lakshmanan, M.: Hyper chaos in a modified canonical Chua's circuit. Int. J. Bifurcation and Chaos 14, 221–243 (2004)
13. Wang, J.Z., Duan, Z.S., Huang, L.: Dichotomy of nonlinear systems: Application to chaos control of nonlinear electronic circuit. Phys. Lett. A 351, 143–152 (2006)
14. Liu, X., Wang, J., Huang, L.: Attractors of fourth-order chua's circuit and chaos control. Int. J. Bifurcation and Chaos 17(8), 2705–2722 (2007)
15. Yin, Y.Z.: Synchronization of chaos in a modified Chua's circuit using continuous control. Int. J. Bifurcation and Chaos 11, 2101–2117 (1996)

A Phase Reduction Method for Weakly Coupled Stochastic Oscillator Systems

Akihisa Ichiki[1] and Yasuomi D. Sato[2,3]

[1] Department of Physics, Faculty of Science, Tokyo Institute of Technology
2-12-1 Ohokayama, Meguro-ku, Tokyo, 152-8551, Japan
aichiki@mikan.ap.titech.ac.jp

[2] Department of Brain Science and Engineering, Kyushu Institute of Technology
2-4 Hibikino, Wakamatsu-ku, Kitakyushu, 808-0196, Japan
sato-y@brain.kyutech.ac.jp

[3] Frankfurt Institute for Advanced Studies (FIAS), Johann Wolfgang Goethe
University Ruth-Moufang-Str. 1, 60438,
Frankfurt am Main, Germany
sato@fias.uni-frankfurt.de

Abstract. We progressively propose a generalized phase reduction method for a stochastic system of weakly coupled oscillators, regardless of the noise intensity, and analyze dynamical behavior in such a system. This is because noise effects on a phase space were so far described for an uncoupled single stochastic oscillator, subjected only to weak noise. Our method is established with definition of "a phase of the distribution of state variables," rather than by defining distributions on a phase space. The stochastic system of weakly coupled oscillators can then be reduced straightforward to one dimensional phase dynamics. It is also confirmed that our method can be applied into the deterministic system without any noise intensity.

Keywords: a phase reduction method, weakly couplings, stochastic oscillators, noise independence.

1 Introduction

In this paper, we aim at giving a more progressive form of the phase reduction method, which is more practicable even for a weakly coupled system of stochastic oscillators, compared to the typical phase reduction method [1,2]. The conventional use of the phase reduction method for stochastic systems [3,4] has long fallen under the following two restrictions: (i) the weak noise and (ii) the uncoupling between stochastic oscillators. It will thus be necessary and important to construct the generalized phase reduction method beyond these restrictions because a diverse number of physical systems exhibiting oscillatory dynamics can be regarded to be not only under weak noise, but also under strong one. For example, in physiological experiments using hippocampal formation slices, synchronous behaviors in a pair of neurons via chemical synapses were observed amidst very strong environmental noise [5].

© Springer-Verlag Berlin Heidelberg 2011

A typical phase reduction method [1,2] is a powerful tool for straightforwardly reducing the dynamics of a nonlinear N-dimensional oscillator to the simple one-dimensional phase dynamics in the deterministic system. This method is also widely employed for oscillatory systems of electrical circuits [6] and brain waves [7]. In recent years, the phase reduction method has been, in turn, applied to an uncoupled stochastic oscillator influenced only by weak noise as a small perturbation [8,9]. This approach is expected to be useful for studying the phase description of infinitesimal changes in noise intensity for a single oscillator. Yoshimura and Arai pointed out that the natural frequency of the single oscillator varies with the effect of noise [3]. Their approach and results may be interesting.

Here, since we are interested in studying the way in which two or more interacting stochastic oscillators behave, independent of noise intensity, we must consider the possibility that the behavior of coupled or uncoupled oscillators remains periodic even with finite noise intensity [10]. The validity of the phase reduction method is required for coupled stochastic oscillators with a wide range of noise intensity. In order to construct the innovative phase reduction method applicable to weakly coupled stochastic oscillator systems, regardless of the noise intensity, in our approach, a phase variable is defined as on a space of the probability density of the state variable (see a right figure in Fig. 1) as long as the uncoupled oscillators show statistically periodic behaviors. It is noticed that a Stuart-Landau oscillator with multiplicative colored noise is one of the easiest models to which our approach is applicable. This model is given by $dx = [x/\tau_1 - c_0 y - x(x^2+y^2)/r^2\tau_2]dt + x d\rho/(x^2+y^2)^{1/2}$, $dy = [c_0 x + y/\tau_1 - y(x^2+y^2)/r^2\tau_2]dt + y d\rho/(x^2+y^2)^{1/2}$, $d\rho = -\gamma\rho dt + b dW$, where t_1, t_2, c_0, r, b are constants and dW is a Wiener process. Intuitively, one might expect that it obviously reaches a temporally periodic state, even with the large noise intensity. Thus, the noise influence on the system can no longer be a perturbation. Weak couplings between stochastic oscillators can only be regarded as small perturbations. On the contrary, in recent publications [8,9], these phase variables are frequently defined within the well-known framework of the state space (see a left figure in Fig. 1), because the phase reduction method for this case [3] can be done using the perturbation expansion in terms of the small noise.

2 Phase Reduction Method

In this work, a system of two weakly coupled N-dimensional oscillators subjected to white noise is considered as the general form of the Stratonovich stochastic differential equation:

$$\dot{x}_i^{(n)} = F_i^{(n)}(\boldsymbol{x}^{(n)}) + \epsilon I_i^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})}) + G_i^{(n)}(\boldsymbol{x}^{(n)})\xi_i^{(n)}(t) \tag{1}$$

where $\boldsymbol{x}^{(n)} = (x_1^{(n)}, \cdots, x_N^{(n)}) \in \mathbb{R}^N$ is the state variable of the N-dimensional oscillator n $(n = 1, 2)$. $\boldsymbol{F}^{(n)}$ is the vector field of the uncoupled deterministic

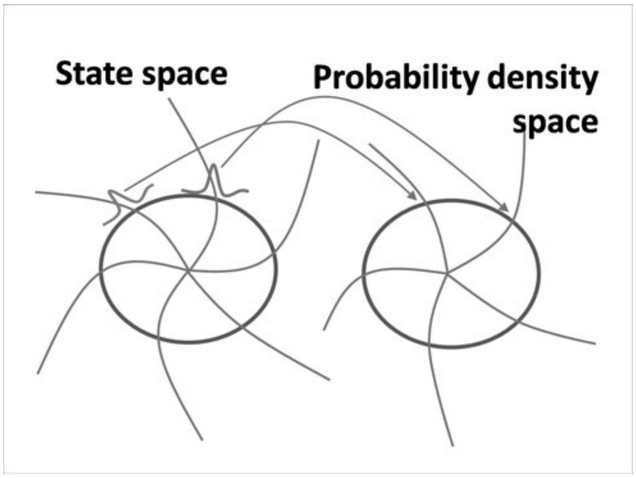


Fig. 1. Schematic images of our phase reduction. The "phase variable" defined in a state space statistically distributes (left circle). The probability density of the "phase variable" is temporally periodic. On the other hand, our phase variable is defined in a probability density space (right circle). Our phase variable corresponds to the probability density of the "phase variable" defined in a state space.

dynamics. The vector function $\boldsymbol{G}^{(n)}$ plays the role of noise intensity. $\boldsymbol{I}^{(n)}$ is the coupling function of the states of two oscillators $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$, and $\bar{n}$ is the counterpart of n. $\xi_i^{(n)}(t)$ is the white Gaussian noise such that $\langle \xi_i^{(n)}(t) \rangle = 0$ and $\langle \xi_i^{(n)}(t)\xi_j^{(m)}(s) \rangle = 2\delta_{ij}\delta_{nm}\delta(t-s)$ where $\langle \ldots \rangle$ denotes averaging over the realization of ξ and δ is the Dirac delta function. We call the constant $D > 0$ the noise intensity. Similar to the well-known phase reduction method used previously, we will restrict ourselves to instances in which coupling strength $|\epsilon|$ sufficiently small. It is noticed here that, if the vector function $\boldsymbol{G}^{(n)}$ has no zero components, according to the H-theorem [11], an uncoupled stochastic oscillator system tends to a steady state and the periodic state does not appear. Thus, we assume $\boldsymbol{G}^{(n)}$ has some zero components, because of the presence of the periodic state under the influence of noise.

2.1 Uncoupled Stochastic Oscillator

An uncoupled individual stochastic oscillator n under our aforementioned assumption may obey the periodic probability density $P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, t)$ with its period $T^{(n)}$ as the limit state solution for the Fokker-Planck equation

$$\frac{\partial P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, t)}{\partial t} = \hat{L}^{(n)} P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, t), \tag{2}$$

$$\hat{L}^{(n)} = -\sum_{i=1}^{N} \frac{\partial}{\partial x_i^{(n)}} \left[F_i^{(n)} + G_i^{(n)} \frac{\partial G_i^{(n)}}{\partial x_i^{(n)}} \right] + \sum_{i=1}^{N} \frac{\partial^2}{\partial x_i^{(n)2}} \left(G_i^{(n)} \right)^2, \tag{3}$$

where $\hat{L}^{(n)}$ is the Fokker-Planck operator for the uncoupled stochastic oscillator n. Let us consider that each stochastic oscillator exhibits periodic statistical properties under the influence of noise, and each of the weak coupled oscillators also follows a temporally periodic probability density close to that of the uncoupled case.

We briefly discuss the reason for taking a general form of the stochastic oscillator system such as Eqs. (2) and (3), in which each of the dynamic variables is influenced by a different noise source. In physiologically plausible spiking models of the Hodgkin-Huxley [12] and Morris-Lecar [13] types, the state variables in their dynamics must be defined as representing changes in the membrane potential, or controlling the opening and closing of ionic channels. Each of these entities is affected by a different noise source, in terms of modeling a realistic system [14]. Nevertheless, in [3,4], many researchers studying a stochastic phase reduction method dealt with the stochastic system consisting of one N-dimensional oscillator:

$$\dot{\boldsymbol{x}} = \boldsymbol{F}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\xi(t).$$

Since this means that all dynamic variables are influenced by the common noise source $\xi(t)$, the method should have ample room for extension to the generalized form of the stochastic oscillator system. Therefore, in this paper, we decided to formulate the phase reduction method for stochastic systems with non-identical noise sources of arbitrary noise intensity.

2.2 Weakly Coupled Stochastic Oscillators

Return to phase reduction studies on coupled stochastic oscillator systems, the Fokker-Planck equation of Eq. (1) is firstly given by

$$\begin{aligned}\frac{\partial P(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, t)}{\partial t} &= \left(\hat{L}^{(1)} + \hat{L}^{(2)}\right) P(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, t) \\ &\quad -\epsilon \sum_{i=1}^{N} \left[\frac{\partial}{\partial x_i^{(1)}} I_i^{(1)} + \frac{\partial}{\partial x_i^{(2)}} I_i^{(2)}\right] P(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, t). \qquad (4)\end{aligned}$$

Then the marginal probability density for each oscillator $P^{(n)}(\boldsymbol{x}^{(n)}, t)$ obeys

$$\begin{aligned}\frac{\partial P^{(n)}(\boldsymbol{x}^{(n)}, t)}{\partial t} &= \hat{L}^{(n)} P^{(n)}(\boldsymbol{x}^{(n)}, t) \\ &\quad -\epsilon \sum_{i=1}^{N} \frac{\partial}{\partial x_i^{(n)}} \int d\boldsymbol{x}^{(\bar{n})} I_i^{(n)} P(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, t), \qquad (5)\end{aligned}$$

where the natural boundary conditions have been used, and the integration along the right hand side is carried out over the state variables of the oscillator $\bar{n}(\neq n)$. The coupling strength $|\epsilon|$ is assumed to be sufficiently small.

The marginal probability density $P^{(n)}(\boldsymbol{x}^{(n)}, t)$, which is assumed to remain periodic for small values of the interaction $\epsilon I_i^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})})$, is decomposed into two types of deviations: phase deviation $\tau_n(t)$ and orbital deviation $\epsilon Q^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t))$. The marginal probability density for each oscillator is thus given by

$$P^{(n)}(\boldsymbol{x}^{(n)}, t) = P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t)) + \epsilon Q^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t)), \tag{6}$$

where $P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t))$ is the component parallel to the periodic solution of the uncoupled system. $\epsilon Q^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t))$ is a component in the orbital deviation in the space of probability density. Assume that $\dot{\tau}_n = \mathcal{O}(\epsilon)$ provides a good approximation for the marginal probability density [1]. The function $\tau_n(t)$ represents the phase advance or phase delay induced by the weak interaction.

In addition, we use an approximation for probability density, equivalently written as

$$P(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, t) = P_{\mathrm{p}}^{(1)}(\boldsymbol{x}^{(1)}, t + \tau_1(t)) P_{\mathrm{p}}^{(2)}(\boldsymbol{x}^{(2)}, t + \tau_2(t)) + \mathcal{O}(\epsilon).$$

This means that the distributed states between the two oscillators are almost independent. Consequently, Eq. (5) is rewritten as

$$\begin{aligned}\dot{\tau}_n(t) \left.\frac{\partial P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, s)}{\partial s}\right|_{s=t+\tau_n(t)} = & -\epsilon \sum_{i=1}^{N} \frac{\partial}{\partial x_i^{(n)}} \int d\boldsymbol{x}^{(\bar{n})} I_i^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})}) \\ & \times P_{\mathrm{p}}^{(1)}(\boldsymbol{x}^{(1)}, t + \tau_1(t)) P_{\mathrm{p}}^{(2)}(\boldsymbol{x}^{(2)}, t + \tau_2(t)) \\ & + \epsilon \hat{L}^{(n)} Q^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t)) \\ & - \epsilon \left.\frac{\partial Q^{(n)}(\boldsymbol{x}^{(n)}, s)}{\partial s}\right|_{s=t+\tau_n(t)}. \end{aligned} \tag{7}$$

where $\dot{\tau}_n(t)$ is constant with respect to time variable t (at least, up to order $\mathcal{O}(\epsilon)$), according to our assumption of Eq. (6). This is because the modulated period for each coupled oscillator $\tilde{T}^{(n)}$ is given by $T^{(n)} = \tau_n(t + \tilde{T}^{(n)}) - \tau_n(t)$ for arbitrary t. This condition does indeed hold for sufficiently small $|\epsilon|$.

We introduce the function $R^{(n)}(\boldsymbol{x}, t)$ defined by

$$R^{(n)}(\boldsymbol{x}, t) = \frac{\partial P_{\mathrm{p}}^{(n)}(\boldsymbol{x}, t)}{\partial t} \bigg/ \int d\boldsymbol{x} \left(\frac{\partial P_{\mathrm{p}}^{(n)}(\boldsymbol{x}, t)}{\partial t} \right)^2, \tag{8}$$

which is tangent to the periodic limit state solution $P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, t)$. This may be related to a necessary orthogonal condition to obtain a one-dimensional phase equation using the deterministic phase reduction method[2,6]. In what follows, let Eq. (8) be also satisfied with the orthogonal condition to the velocity of distributed state motion. Notice that $Q^{(n)}$ appears in Eq. (7) as the orbital effect of the weak interaction $\epsilon I^{(n)}$. Furthermore, the phase deviation component of the vector field near the point $P_{\mathrm{p}}^{(1)} P_{\mathrm{p}}^{(2)}$ in probability density space is same as that at $P_{\mathrm{p}}^{(1)} P_{\mathrm{p}}^{(2)}$ up to order unity. Then, the dynamics of $\tau_n(t)$ are given by

$$\dot{\tau}_n = \epsilon \int d\boldsymbol{x}^{(n)} \sum_{i=1}^{N} \frac{\partial R^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t))}{\partial x_i^{(n)}}$$

$$\times \left[\int d\boldsymbol{x}^{(\bar{n})} I_i^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})}) P_{\mathrm{p}}^{(\bar{n})}(\boldsymbol{x}^{(\bar{n})}, t + \tau_{\bar{n}}(t)) \right]$$
$$\times P_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, t + \tau_n(t)). \tag{9}$$

We finally obtain

$$\frac{d\theta_n}{dt} = \frac{1}{T^{(n)}} + \frac{\epsilon}{T^{(n)}} \int d\boldsymbol{x}^{(n)} \sum_{i=1}^{N} \frac{\partial \tilde{R}^{(n)}(\boldsymbol{x}^{(n)}, \theta_n(t))}{\partial x_i^{(n)}}$$
$$\times \left[\int d\boldsymbol{x}^{(\bar{n})} I_i^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})}) \tilde{P}_{\mathrm{p}}^{(\bar{n})}(\boldsymbol{x}^{\bar{n}}, \theta_{\bar{n}}(t)) \right]$$
$$\times \tilde{P}_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, \theta_n(t)), \tag{10}$$

introducing the phase variable of each oscillator $\theta_n(t) = (t+\tau_n(t))/T^{(n)}$. Here we have defined the functions $\tilde{P}_{\mathrm{p}}^{(n)}(\boldsymbol{x}, \theta_n(t)) = P_{\mathrm{p}}^{(n)}(\boldsymbol{x}, t+\tau_n(t))$ and $\tilde{R}^{(n)}(\boldsymbol{x}, \theta_n(t)) = R^{(n)}(\boldsymbol{x}, t + \tau_n(t))$.

3 Analysis of Synchronization

In order to study the synchronization phenomena of two coupled oscillators on a one-dimensional phase space, the phase difference between the two oscillators $\phi(t) = \theta_2(t) - \theta_1(t)$ is of interest. It is noticed that $\theta_n(t)$ is split into fast and slow variables, since the time dependence of the second term on the right hand side of Eq. (10) is small ($\mathcal{O}(\epsilon)$). Averaging the fast phase variable in Eq. (10) yields

$$\frac{d\phi}{dt} = \frac{1}{T^{(2)}} - \frac{1}{T^{(1)}} + \epsilon H_2(-\phi) - \epsilon H_1(\phi), \tag{11}$$

where $H_n(\phi)$ is defined by

$$H_n(\phi) = \frac{1}{T^{(n)}} \int_0^1 d\theta \int d\boldsymbol{x}^{(n)} \sum_{i=1}^{N} \frac{\partial \tilde{R}^{(n)}(\boldsymbol{x}^{(n)}, \theta)}{\partial x_i^{(n)}}$$
$$\times \left[\int d\boldsymbol{x}^{(\bar{n})} I_i^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})}) \tilde{P}_{\mathrm{p}}^{(\bar{n})}(\boldsymbol{x}^{(\bar{n})}, \theta + \phi) \right] \tilde{P}_{\mathrm{p}}^{(n)}(\boldsymbol{x}^{(n)}, \theta), \tag{12}$$

where it is noticed that if the periods of the two oscillators $T^{(1)}$ and $T^{(2)}$ satisfy $T^{(2)} - T^{(1)} = \mathcal{O}(\epsilon)$, the phase advance or phase delay per unit time, i.e., $\dot{\phi}(t)$, is small ($\mathcal{O}(\epsilon)$), and the phase difference between the two oscillators can be treated as a constant during the period $T^{(n)}$. In this case, one can average the fast phase variable in Eq. (10). This fact is consistent even with our assumption that the marginal probability density $P^{(n)}$, i.e., the solution of Eq. (6), remains periodic. However if the difference of the periods of the two oscillators is of order unity, i.e., $T^{(2)} - T^{(1)} = \mathcal{O}(\epsilon)$, such an averaging process is invalid. In this case, the marginal probability density $P_{\mathrm{p}}^{(n)}(\boldsymbol{x}, t + T^{(n)}(t))$ at time $t + T^{(n)}(t)$ differs from the one at time t and the averaging procedure during the period $T^{(n)}$ becomes invalid.

Eq. (11) takes the same form as the dynamics for the phase difference in the deterministic systems, because it is straightforward to see that Eq. (11) reproduces the well-known phase difference dynamics for the deterministic system, when $P(\boldsymbol{x}^{(n)}, t) = \delta(\boldsymbol{x}^{(n)} - \boldsymbol{x}_{\mathrm{p}}^{(n)}(t))$, where $\boldsymbol{x}_{\mathrm{p}}^{(n)}(t)$ is the periodic solution of the deterministic dynamics. From Eq. (11), we can obtain a great deal of information regarding the synchronization of the system. First, let us consider the case where the periods of each oscillator are identical as a consequence of the weak interaction. This is regarded as the synchronization phenomenon of the most simple case. Here, the phase difference $\phi(t)$ is constant with respect to time, i.e., $d\phi/dt = 0$. Similar to the well-known deterministic case, it is expected that the function $H_2(-\phi) - H_1(\phi)$ is a bounded function with its upper bound $\omega_{\max}$ and lower bound $\omega_{\min}$. Under these circumstances, the condition for the synchronization is given by $\epsilon\omega_{\min} < \Delta T/(T^{(1)})^2 < \epsilon\omega_{\max}$, where $\Delta T = T^{(2)} - T^{(1)}$. Furthermore, the stationary solution of the phase difference ϕ_∞ is given by $0 = \epsilon H_2(-\phi_\infty) - \epsilon H_1(\phi_\infty) - \Delta T/(T^{(1)})^2$. Then the modulated period under the influence of the weak coupling is given by $\tilde{T}^{(1)} = \tilde{T}^{(2)} = T^{(1)} - \epsilon(T^{(1)})^2 H_1(\phi_\infty)$.

One may wonder whether there is the possibility that the oscillator 2 oscillates through m_2 cycles over time it takes oscillator 1 to complete m_1 cycles ($(m_1, m_2) \in \mathbb{N}^2$). This is also regarded as synchronization. In this case, the ratio of the periods $\tilde{T}^{(1)}$ and $\tilde{T}^{(2)}$ of the two oscillators modulated by the weak interaction becomes rational. Since we have assumed that $\tilde{T}^{(n)}$ is constant with respect to time up to $\mathcal{O}(\epsilon)$, we have $\tilde{T}^{(n)} = T^{(n)} - \epsilon(T^{(n)})^2 H_n(\delta_{n,1}\phi - \delta_{n,2}\phi) + \mathcal{O}(\epsilon^2)$ from Eq. (10). Then, if such synchronization occur, there exists the set of natural numbers (m_1, m_2) satisfying $m_1\tilde{T}^{(1)} = m_2\tilde{T}^{(2)}$. The set of minimum numbers (m_1, m_2) satisfying this condition gives the synchronization period of the total system, i.e., $T_{\mathrm{tot}} = m_1\tilde{T}^{(1)} = m_2\tilde{T}^{(2)}$. However, such natural numbers m_1 and m_2 are very large, i.e.,$m_1 = \mathcal{O}(1/\epsilon)$ and $m_2 = \mathcal{O}(1/\epsilon)$. This means that the synchronization period T_{tot} is of order $\mathcal{O}(1/\epsilon)$. In this time scale, our assumption that $\tilde{T}^{(n)}$ is constant with respect to time becomes invalid. In order to deal with this type of synchronization, it is necessary to take into account higher order corrections of ϵ.

4 Discussion and Conclusion

Once the temporally periodic probability density for each oscillator $P_{\mathrm{p}}^{(n)}$ is obtained by some methods, e.g., numerical simulation or solving the Fokker-Planck equation (4) numerically, the function $R^{(n)}$ can be calculated by Eq. (8). Then Eq. (11) can be solved in principle. The time evolution of the phase difference $\phi(t)$ under the influence of the weak interaction is given by Eq. (11), and the shift in the frequency of each oscillator is given by Eq. (10).

The phase reduction method formulated by using higher order theory will be reported elsewhere. Higher order theory would provide insight on into noise-induced synchronization. Noise-induced synchronization for weakly coupled two oscillator systems may be characterized into two types. The first type of noise-induced synchronization is for the stability change of the marginal probability density $P^{(n)}(\boldsymbol{x}^{(n)}, t)$. In this case, a non-periodic stable solution for Eq. (6) becomes unstable when subject to a slight changes in noise intensity, causing a new periodic stable solution to appear. Since such a periodic solution cannot be approximated by the non-periodic solution before the change in noise intensity, this situation is beyond the dynamical phase description of our approach. The second type of noise-induced synchronization is for the change in the stability of the stationary solution for the dynamics of phase difference $\phi(t)$. In this case, the probability density $P(\boldsymbol{x}^{(n)}, t)$ varies slightly in response to the effects of small changes in noise intensity. This change in the marginal probability density $P(\boldsymbol{x}^{(n)}, t)$ causes an infinitesimal change in the form of the function $H_2(-\phi) - H_1(\phi)$, and then the stability of the stationary solution for Eq. (11) may also change. The scenario for noise-induced synchronization could be treated within the framework of the phase reduction method. However, for investigating such a kind of noise-induced synchronization, we must formulate an extended version of our phase reduction method involving the higher order of ϵ, since small changes in the function $H_2(-\phi) - H_1(\phi)$ are expected to be $\mathcal{O}(\epsilon)$.

In this paper, we have presented a phase reduction method valid for stochastic oscillator systems with a wide range of noise intensities, and have constructed the phase reduction method regarding the weak interaction as a perturbation. The main ideas of our approach are (i) definition of the phase variables in the space of probability densities, (ii) decomposition of small perturbations, applicable even for a stochastic oscillator system. These ideas are valid even for the case where small changes in noise intensity are regarded as a perturbation. This method has shown that the analysis of the synchronization of coupled stochastic oscillators is relatively analogous to the analysis of the phase reduction for the deterministic case.

Acknowledgments

A. I. was supported by the Grant-in-Aid for JSPS Fellows No. 20.9513. Y.D.S. was financially supported by the Grant-in-Aid for Young Scientist (B) No. 22700237.

References

1. Kuramoto, Y.: Chemical Oscillation, Waves, and Turbulence. Springer, Tokyo (1984)
2. Sato, Y.D.: Doctoral Thesis, Synchronization Phenomena in a Pair of Coupled Neural Oscillator Systems. Tokyo Institute of Technology, Tokyo (2005)
3. Yoshimura, K., Arai, K.: Phase Reduction of Stochastic Limit Cycle Oscillators. Phys. Rev. Lett. 101, 154101 (2008)
4. Teramae, J., Tanaka, D.: Robustness of the noise-induced phase synchronization in a general class of limit cycle oscillators. Phys. Rev. Lett. 93, 204103 (2004)
5. Netoff, T., Matthew, B., Alan, D.: Synchronization in Hybrid Neuronal Networks of the Hippocampal Formation. J. Neurophysiol. 93, 1197–1208 (2005)

6. Mehrotra, A., Sangiovanni-Vincentelli, A.: Noise Analysis of Radio Frequency Circuits. Kluwer Academic Publishers, Dordrecht (2004)
7. Perez Velazquez, J.L., Galán, R.F., Garcia Dominguez, L., Leshchenko, Y., Lo, S., Belkas, J., Guevara Erra, R.: Phase response curves in the characterization of epileptiform activity. Phys. Rev. 76, 061912 (2007)
8. Teramae, J., Nakao, H., Ermentrout, G.B.: Stochastic Phase Reduction for a General Class of Noisy Limit Cycle Oscillators. Phys. Rev. Lett. 102, 194102 (2009)
9. Yoshimura, Y.: Phase Reduction of Stochastic Limit-Cycle Oscillators. In: Schuster, H.G. (ed.) Reviews of Nonlinear Dynamics and Complexity, pp. 59–90. WILEY-VCH Verlag GmbH & Co., KGaA (2010)
10. Ichiki, A., Ito, H., Shiino, M.: Chaos-nonchaos phase transitions induced by multiplicative noise in ensembles of coupled two-dimensional oscillators. Physica E 40, 402–405 (2007)
11. Gardiner, C.W.: Handbook of Stochastic Methods. Springer, New York (1997)
12. Hodgkin, A.L., Huxley, A.F.: A quantitative description of membrane current and its application to conduction and excitation in nerve. J. Physiol. 117, 500–544 (1952)
13. Morris, C., Lecar, H.: Voltage oscillations in the barnacle giant muscle fiber. Biophys. J. 35, 193–213 (1981)
14. Fox, R.F.: Stochastic versions of the Hodgkin-Huxley equations. Biophys. J. 72, 2068–2074 (1997)

Anti-periodic Solutions for High-Order Neural Networks with Mixed Time Delays

Xiaofeng Chen and Qiankun Song

Department of Mathematics, Chongqing Jiaotong University,
Chongqing 400074, China
Qiankunsong@163.com

Abstract. In the paper, the problem on the stability of anti-periodic solutions is investigated for high-order neural networks with discrete and distributed time delays. Several sufficient conditions for checking the existence, uniqueness and global exponential stability of anti-periodic solution for the considered neural networks are given. A numerical example is given to show its effectiveness.

Keywords: Anti-periodic, Hight-order Neural networks, Exponential stability, Mixed delays.

1 Introduction

The study of anti-periodic solutions for nonlinear evolution equations is closely related to the study of periodic solutions, and it was initiated by Okochi [1]. It is well known that the existence of anti-periodic solutions play a key role in characterizing the behavior of nonlinear differential equations [2, 3]. During the past twenty years anti-periodic problems of nonlinear differential equations have been extensively studied by many authors, for example, see [4–9] and references therein. In addtion, anti-periodic trigonometric polynomials are important in the study of interpolation problems [10, 11], and anti-periodic wavelets are discussed in [12].

Moreover, due to the fact that 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, for example, see [13–16] and references therein. However, to the best of our knowledge, very few results are available on the existence and exponential stability of anti-periodic solutions for high-order neural networks.

Motivated by the above discussions, in the paper, we discuss the existence and exponential stability of anti-periodic solutions for high-order neural networks with discrete and distributed time delays which can be described by the following delay differential equations:

$$\dot{x}_i(t) = -\, b_i(t)x_i(t) + \sum_{j=1}^{n} c_{ij}(t)g_j(x_j(t-\tau_{ij}(t)))$$

© Springer-Verlag Berlin Heidelberg 2011

$$+\sum_{j=1}^{n} d_{ij}(t)\int_0^{\sigma} k_{ij}(s)g_j(x_j(t-s))ds$$
$$+\sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t)g_j(x_j(t-u_{jl}(t)))g_l(x_l(t-v_{jl}(t)))+I_i(t), \qquad (1)$$

where $i=1,2,\cdots,n$, $x_i(t)$ denotes the state of the ith unit, $b_i(t)>0$ denotes the passive decay rate, $c_{ij}(t)$, $d_{ij}(t)$ and $e_{ijl}(t)$ are the synaptic connections strengths, $\tau_{ij}(t)\geq 0$, $u_{ij}(t)\geq 0$ and $v_{ij}(t)\geq 0$ correspond to the delays, $I_i(t)$ denotes the external inputs, g_j is the activation function of neurons, and the delay kernel $k_{ij}(s)$ is real value negative continuous functions defined on $\mathbb{R}^+ := [0,\infty)$.

Let $u(t)\in C(\mathbb{R},\mathbb{R})$. $u(t)$ is said to be w-anti-periodic on $\mathbb{R}$ if

$$u(t+w)=-u(t), \quad \forall t\in\mathbb{R},$$

where w is a positive constant. For convenience, we introduce some notations

$$\overline{c}_{ij}=\sup_{t\in\mathbb{R}}|c_{ij}(t)|, \quad \overline{d}_{ij}=\sup_{t\in\mathbb{R}}|d_{ij}(t)|, \quad \overline{e}_{ijl}=\sup_{t\in\mathbb{R}}|e_{ijl}(t)|,$$
$$\overline{\tau}=\max_{1\leq i,j\leq n}\left\{\sup_{t\in\mathbb{R}}\tau_{ij}(t)\right\}, \quad \overline{u}=\max_{1\leq j,l\leq n}\left\{\sup_{t\in\mathbb{R}}u_{jl}(t)\right\},$$
$$\overline{v}=\max_{1\leq j,l\leq n}\left\{\sup_{t\in\mathbb{R}}v_{jl}(t)\right\}, \quad \tau=\max\left\{\overline{\tau},\overline{u},\overline{v},\sigma\right\},$$
$$\overline{I}_i=\sup_{t\in\mathbb{R}}|I_i(t)|, \quad I=\max_{1\leq i\leq n}\left\{\overline{I}_i\right\}$$

For $\varphi(t)=(\varphi_1(t),\varphi_2(t),\cdots,\varphi_n(t))\in C([-\tau,0],\mathbb{R}^n)$, we define the norm

$$\|\varphi\|=\max_{1\leq i\leq n}\left\{\sup_{t\in[-\tau,0]}|\varphi_i(t)|\right\}.$$

Throughout this paper, it will be assumed that

(H_1) For $i,j,l=1,2,\cdots,n$, $b_i,c_{ij},d_{ij},e_{ijl},g_j\in C(\mathbb{R},\mathbb{R})$, $k_{ij}\in C(\mathbb{R}^+,\mathbb{R}^+)$, τ_{ij}, $u_{jl},v_{jl}\in C(\mathbb{R},\mathbb{R}^+)$, $\sigma\in\mathbb{R}^+$, and

$$b_i(t+w)=b_i(t), \quad I_i(t+w)=-I_i(t), \quad \forall t\in\mathbb{R},$$
$$c_{ij}(t+w)g_j(u)=-c_{ij}(t)g_j(-u), \quad \forall t,u\in\mathbb{R},$$
$$d_{ij}(t+w)g_j(u)=-d_{ij}(t)g_j(-u), \quad \forall t,u\in\mathbb{R},$$
$$e_{ijl}(t+w)g_j(u)g_l(u)=-e_{ijl}(t)g_j(-u)g_l(-u), \quad \forall t,u\in\mathbb{R},$$
$$\tau_{ij}(t+w)=\tau_{ij}(t), \quad u_{jl}(t+w)=u_{jl}(t), \quad v_{jl}(t+w)=v_{jl}(t), \quad \forall t\in\mathbb{R}.$$

(H_2) For $i=1,2,\cdots,n$, there exists a positive constant $\underline{b}_i$ such that

$$b_i(t)\geq\underline{b}_i, \quad \forall t\in\mathbb{R}.$$

(H_3) For $j = 1, 2, \cdots, n$, there exist positive constants M_j and L_j such that

$$g_j(0) = 0, \quad |g_j(u)| \leq M_j, \quad |g_j(u) - g_j(v)| \leq L_j|u - v|, \quad \forall u, v \in \mathbb{R}.$$

(H_4) For $i, j = 1, 2, \cdots, n$, $p_{ij}(\delta) := \int_0^\sigma e^{\delta s} k_{ij}(s)ds$ is continuous on $[0, \epsilon)$, where $\epsilon > 0$ and $p_{ij}(0) \leq 1$.

(H_5) For all $i = 1, 2, \cdots, n$, there exists a constant $\eta > 0$ such that

$$-\underline{b}_i + \sum_{j=1}^{n}(\overline{c}_{ij} + \overline{d}_{ij})L_j + \sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}\Big(L_jM_l + L_lM_j\Big) < -\eta < 0.$$

2 Preliminaries

In this section, we recall some definitions and make some preparations.

Definition 1. *Let $x^*(t) = (x_1^*(t), x_2^*(t), \cdots, x_n^*(t))$ be the solution of system (1) with initial value $\varphi^* \in C([-\tau, 0], \mathbb{R}^n)$. If there exist constants $\lambda > 0$ and $N > 1$ such that for any solution $x(t) = (x_1(t), x_2(t), \cdots, x_n(t))$ of system (1) with initial value $\varphi \in C([-\tau, 0], \mathbb{R}^n)$ satisfies*

$$|x_i(t) - x_i^*(t)| \leq N\|\varphi - \varphi^*\|e^{-\lambda t}, \quad \forall t > 0, i = 1, 2, \cdots, n.$$

Then $x^*(t)$ is said to be globally exponentially stable.

Lemma 1. *Suppose that (H_1)-(H_5) hold. Let $x(t) = (x_1(t), x_2(t), \cdots, x_n(t))$ be a solution of system (1) with initial conditions*

$$x_i(s) = \varphi_i(s), \quad |\varphi_i(s)| < \gamma, \quad s \in [-\tau, 0],$$

where $\gamma > \frac{I}{\eta}$. Then for all $i = 1, 2, \cdots, n$,

$$|x_i(t)| < \gamma, \quad t \in [0, +\infty). \tag{2}$$

Proof. Assume, by way of contradiction, that (2) does not hold. Then there must exist $i \in \{1, 2, \cdots, n\}$ and the first time $t_1 > 0$ such that

$$|x_i(t_1)| = \gamma, \quad |x_i(t)| < \gamma, \quad \forall t \in [-\tau, t_1),$$
$$|x_j(t)| < \gamma, \quad \forall t \in [-\tau, t_1], \quad j \neq i, \quad j = 1, 2, \cdots, n.$$

Calculating the upper left derivative of $|x_i(t_1)|$, together with , we can obtain

$$D^+(|x_i(t_1)|) \leq -b_i(t_1)|x_i(t_1)| + \sum_{j=1}^{n}\overline{c}_{ij}L_j|x_j(t_1 - \tau_{ij}(t_1))|$$
$$+ \sum_{j=1}^{n}\overline{d}_{ij}L_j\int_0^\sigma k_{ij}(s)|x_j(t_1 - s)|ds$$

$$
\begin{aligned}
&+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}L_jL_l|x_j(t_1-u_{jl}(t_1))||x_l(t_1-v_{jl}(t_1))|+\overline{I}_i,\\
&\le\left[-\underline{b}_i+\sum_{j=1}^{n}(\overline{c}_{ij}+\overline{d}_{ij})L_j+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}L_jM_l\right]\gamma+\overline{I}_i\\
&<-\eta\frac{I}{\eta}+\overline{I}_i\le 0,
\end{aligned}
$$

which is a contradiction to $D^+(|x_i(t_1)|)\ge 0$ and implies that (2) holds.

Remark 1. In view of the boundedness of this solution, from the theory of functional differential equations in [17], it follows $x^*(t)$ can be defined on $[-\tau,+\infty)$, provided that the initial conditions are bounded by γ.

Lemma 2. *Suppose that (H_1)-(H_5) hold. Let $x^*(t)=(x_1^*(t),x_2^*(t),\cdots,x_n^*(t))$ be the solution of system (1) with initial value $\varphi^*=(\varphi_1^*(t),\varphi_2^*(t),\cdots,\varphi_n^*(t))$ being bounded by γ. Then $x^*(t)$ is globally exponentially stable.*

Proof. From (H_5), we can choose a small enough positive constant λ such that

$$
\lambda-\underline{b}_i+\sum_{j=1}^{n}(\overline{c}_{ij}e^{\lambda\tau}+\overline{d}_{ij}p_{ij}(\lambda))L_j+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}\Big(L_jM_l+L_lM_j\Big)e^{\lambda\tau}<0,
$$

where $i=1,2,\cdots,n$. Let $x(t)=(x_1(t),x_2(t),\cdots,x_n(t))$ be an arbitrary solution of system (1) with initial value $\varphi=(\varphi_1(t),\varphi_2(t),\cdots,\varphi_n(t))$. Let $y(t)=x(t)-x^*(t)$. Then from (1) we have

$$
\begin{aligned}
\dot{y}_i(t)=&-b_i(t)y_i(t)\\
&+\sum_{j=1}^{n}c_{ij}(t)\Big[g_j(x_j(t-\tau_{ij}(t)))-g_j(x_j^*(t-\tau_{ij}(t)))\Big]\\
&+\sum_{j=1}^{n}d_{ij}(t)\int_0^{\sigma}k_{ij}(s)\Big[g_j(x_j(t-s))-g_j(x_j^*(t-s))\Big]ds\\
&+\sum_{j=1}^{n}\sum_{l=1}^{n}e_{ijl}(t)\Big[g_j(x_j(t-u_{jl}(t)))g_l(x_l(t-v_{jl}(t)))\\
&-g_j(x_j^*(t-u_{jl}(t)))g_l(x_l^*(t-v_{jl}(t)))\Big].
\end{aligned}
\tag{3}
$$

Consider the Lyapunov function $V=(V_1,V_2,\cdots,V_n)$, where

$$
V_i(t)=|y_i(t)|e^{\lambda t},\quad i=1,2,\cdots,n. \tag{4}
$$

Calculating the upper left derivative of $V_i(t)$, by (3) and (4), we obtain

$$
D^+(V_i(t))\le e^{\lambda t}\Bigg\{\lambda|y_i(t)|-\underline{b}_i|y_i(t)|+\sum_{j=1}^{n}\overline{c}_{ij}L_j|y_j(t-\tau_{ij}(t))|
$$

$$
\begin{aligned}
&+\sum_{j=1}^{n}\overline{d}_{ij}L_j\int_0^{\sigma}k_{ij}(s)|y_j(t-s)|ds\\
&+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}\Big[\big|g_j(x_j(t-u_{jl}(t)))g_l(x_l(t-v_{jl}(t)))\\
&-g_j(x_j^*(t-u_{jl}(t)))g_l(x_l(t-v_{jl}(t)))\big|\\
&+\big|g_j(x_j^*(t-u_{jl}(t)))g_l(x_l(t-v_{jl}(t)))\\
&-g_j(x_j^*(t-u_{jl}(t)))g_l(x_l^*(t-v_{jl}(t)))\big|\Big]\Bigg\}\\
&\le e^{\lambda t}\Bigg\{\lambda|y_i(t)|-\underline{b}_i|y_i(t)|+\sum_{j=1}^{n}\overline{c}_{ij}L_j|y_j(t-\tau_{ij}(t))|\\
&+\sum_{j=1}^{n}\overline{d}_{ij}L_j\int_0^{\sigma}k_{ij}(s)|y_j(t-s)|ds\\
&+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}\Big[L_jM_l\big|y_j(t-u_{jl}(t))\big|+L_lM_j\big|y_j(t-v_{jl}(t))\big|\Big]\Bigg\} \qquad (5)
\end{aligned}
$$

From (4) we can choose a constant $N>1$ such that

$$V_i(t)=|y_i(t)|e^{\lambda t}<N\|\varphi-\varphi^*\|,\ \text{for } t\in[-\tau,0], i=1,2,\cdots,n.$$

We claim that

$$V_i(t)<N\|\varphi-\varphi^*\|,\ \text{for } t\in(0,+\infty), i=1,2,\cdots,n. \qquad (6)$$

If it is not true, there exist some $i\in\mathcal{N}$ and the first time $t_1>0$ such that

$$
\begin{gathered}
V_i(t_1)=N\|\varphi-\varphi^*\|,\\
V_j(t)<N\|\varphi-\varphi^*\|,\ \text{for } t\in[-\tau,0), j=1,2,\cdots,n.
\end{gathered}
$$

From (5), we obtain

$$
\begin{aligned}
0&\le D^+(V_i(t_1))\\
&\le e^{\lambda t_1}\Bigg\{\lambda|y_i(t_1)|-\underline{b}_i|y_i(t_1)|+\sum_{j=1}^{n}\overline{c}_{ij}L_j|y_j(t_1-\tau_{ij}(t_1))|\\
&+\sum_{j=1}^{n}\overline{d}_{ij}L_j\int_0^{\sigma}k_{ij}(s)|y_j(t_1-s)|ds\\
&+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}\Big[L_jM_l\big|y_j(t_1-u_{jl}(t_1))\big|+L_lM_j\big|y_j(t_1-v_{jl}(t_1))\big|\Big]\Bigg\}\\
&=(\lambda-\underline{b}_i)V_i(t_1)+\sum_{j=1}^{n}\overline{c}_{ij}L_jV_j(t_1-\tau_{ij}(t_1))e^{\lambda\tau_{ij}(t_1)}
\end{aligned}
$$

$$
\begin{aligned}
&+\sum_{j=1}^{n}\overline{d}_{ij}L_j\int_0^{\sigma}k_{ij}(s)V_j(t_1-s)e^{\lambda s}ds\\
&+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}\Big[L_jM_lV_j(t_1-u_{jl}(t_1))e^{\lambda u_{jl}(t_1)}+L_lM_jV_j(t_1-v_{jl}(t_1))e^{\lambda v_{jl}(t_1)}\Big]\\
&\le\Big[(\lambda-\underline{b}_i)+\sum_{j=1}^{n}\overline{c}_{ij}L_je^{\lambda\tau}+\sum_{j=1}^{n}\overline{d}_{ij}L_jp_{ij}(\lambda)\\
&+\sum_{j=1}^{n}\sum_{l=1}^{n}\overline{e}_{ijl}\Big(L_jM_l+L_lM_j\Big)e^{\lambda\tau}\Big]V_i(t_1)<0,
\end{aligned}
$$

which is a contradiction. Hence (6) holds. It follows that

$$|x_i(t)-x_i^*(t)|<N\|\varphi-\varphi^*\|e^{-\lambda t},\quad \forall t\in(0,+\infty), i=1,2,\cdots,n.$$

By Definition 1, we know that $x^*(t)$ is globally exponentially stable.

3 Main Results

The following is our main result.

Theorem 1. *Suppose that (H_1)-(H_5) hold. Then system* (1) *has exactly one w-anti-periodic solution, which is globally exponentially stable.*

Proof. Let $x(t)=(x_1(t),x_2(t),\cdots,x_n(t))$ be the solution of system (1) with initial value

$$x_i(s)=\varphi_i(s),\quad \varphi_i(s)<\gamma,\quad s\in[-\tau,0],\quad i=1,2,\cdots.$$

By Lemma 1, the solution $x(t)$ is bounded and

$$|x_i(t)|<\gamma,\quad \forall t>0,\quad i=1,2,\cdots.$$

From (1) and (H_1), for any natural number k, we have

$$
\begin{aligned}
&\frac{d}{dt}\Big[(-1)^{k+1}x_i(t+(k+1)w)\Big]\\
&=(-1)^{k+1}\Big\{-b_i(t+(k+1)w)x_i(t+(k+1)w)\\
&\quad+\sum_{j=1}^{n}c_{ij}(t+(k+1)w)g_j(x_j(t+(k+1)w-\tau_{ij}(t+(k+1)w)))\\
&\quad+\sum_{j=1}^{n}d_{ij}(t+(k+1)w)\int_0^{\sigma}k_{ij}(s)g_j(x_j(t+(k+1)w-s))ds\\
&\quad+\sum_{j=1}^{n}\sum_{l=1}^{n}e_{ijl}(t+(k+1)w)g_j(x_j(t+(k+1)w-u_{jl}(t+(k+1)w)))
\end{aligned}
$$

$$\begin{aligned}
&\times g_l(x_l(t+(k+1)w - v_{jl}(t+(k+1)w))) + I_i(t+(k+1)w)\Big\} \\
&= -b_i(t)(-1)^{k+1}x_i(t+(k+1)w) \\
&\quad + \sum_{j=1}^{n} c_{ij}(t)g_j((-1)^{k+1}x_j(t+(k+1)w - \tau_{ij}(t))) \\
&\quad + \sum_{j=1}^{n} d_{ij}(t)\int_0^{\sigma} k_{ij}(s)g_j((-1)^{k+1}x_j(t+(k+1)w - s))ds \\
&\quad + \sum_{j=1}^{n}\sum_{l=1}^{n} e_{ijl}(t)g_j((-1)^{k+1}x_j(t+(k+1)w - u_{jl}(t))) \\
&\quad \times g_l((-1)^{k+1}x_l(t+(k+1)w - v_{jl}(t))) + I_i(t). \qquad (7)
\end{aligned}$$

Thus, for any natural number k, $(-1)^{k+1}x_i(t+(k+1)w)$ are the solutions of (1). Then, by Lemma 2, there exists a constant $N > 1$, such that

$$\begin{aligned}
&|(-1)^{k+1}x_i(t+(k+1)w) - (-1)^k x_i(t+kw)| \\
&\quad \le Ne^{-\lambda(t+kw)} \max_{1\le i\le n}\Big\{\sup_{-\tau\le s\le 0}|x_i(s+w)+x_i(s)|\Big\} \\
&\quad \le 2\gamma Ne^{-\lambda(t+kw)}, \qquad (8)
\end{aligned}$$

where $t+kw>0$, $i=1,2,\cdots n$. It is noted that for any natural number p,

$$\begin{aligned}
&(-1)^{p+1}x_i(t+(p+1)w) \\
&\quad = x_i(x) + \sum_{k=0}^{p}\Big[(-1)^{k+1}x_i(t+(k+1)w) - (-1)^k x_i(t+kw)\Big].
\end{aligned}$$

Thus

$$\begin{aligned}
&|(-1)^{p+1}x_i(t+(p+1)w)| \\
&\quad \le |x_i(x)| + \sum_{k=0}^{p}\Big|(-1)^{k+1}x_i(t+(k+1)w) - (-1)^k x_i(t+kw)\Big|. \qquad (9)
\end{aligned}$$

In view of (8), we can choose a sufficiently large constant $K>0$ and a constant $M>0$, such that

$$\Big|(-1)^{k+1}x_i(t+(k+1)w) - (-1)^k x_i(t+kw)\Big| \le M(e^{-\lambda w})^k, \quad k>K, \qquad (10)$$

on any compact set of $\mathbb{R}$. It follows from (9) and (10) that $\big\{(-1)^p x(t+pw)\big\}$ uniformly converges to a continuous function $x^*(t) = (x_1^*(t), x_2^*(t), \cdots, x_n^*(t))$ on any compact set of $\mathbb{R}$.

Now we will show that $x^*(t)$ is the w-anti-periodic solution of system (1). First, $x^*(t)$ is w-anti-periodic, since

$$x^*(t+w) = \lim_{p\to\infty}(-1)^p x(t+w+pw)$$

$$= -\lim_{p\to\infty} (-1)^{p+1} x(t+(p+1)w) = -x^*(t).$$

Second, we prove that $x^*(t)$ is a solution of (1). In fact, together with the continuity of the right side of (1), (7) implies that

$$\left\{ \frac{d}{dt}\Big[(-1)^{p+1} x_i(t+(p+1)w)\Big] \right\}$$

uniformly converges to a continuous function on any compact set of $\mathbb{R}$. Thus, letting $p \to \infty$, we obtain

$$\begin{aligned}\frac{d(x_i^*(t))}{dt} = & -b_i(t)x_i^*(t) + \sum_{j=1}^n c_{ij}(t) g_j(x_j^*(t-\tau_{ij}(t))) \\ & + \sum_{j=1}^n d_{ij}(t)\int_0^\sigma k_{ij}(s) g_j(x_j^*(t-s))ds \\ & + \sum_{j=1}^n \sum_{l=1}^n e_{ijl}(t) g_j(x_j^*(t-u_{jl}(t))) g_l(x_l^*(t-v_{jl}(t))) + I_i(t),\end{aligned}$$

that is, $x^*(t)$ is a solution of (1). Then, by Lemma 2 we can prove that $x^*(t)$ is globally exponentially stable.

4 Numerical Example

Let us consider the following high-order networks with two neurons and mixed delays:

$$\begin{cases} \dot{x}_1(t) = -x_1(t) + \frac{1}{10} g_1(x_1(t-\sin^2 t)) + \frac{1}{16} g_2(x_2(t-7\sin^2 t)) \\ \qquad + \frac{1}{16}|\sin t| \int_0^{10} e^{-s} g_1(x_1(t-s))ds \\ \qquad + \frac{1}{10}|\cos t| \int_0^{10} e^{-s} g_2(x_2(t-s))ds \\ \qquad + \frac{1}{8}(\sin t) g_1(x_1(t-5\sin^2 t)) g_2(x_2(t-2\sin^2 t)) + 4\sin t, \\ \dot{x}_2(t) = -x_2(t) + \frac{1}{16} g_1(x_1(t-\cos^2 t)) + \frac{1}{10} g_2(x_2(t-5\sin^2 t)) \\ \qquad + \frac{1}{10}|\cos t| \int_0^{10} e^{-s} g_1(x_1(t-s))ds \\ \qquad + \frac{1}{16}|\sin t| \int_0^{10} e^{-s} g_2(x_2(t-s))ds \\ \qquad + \frac{1}{4}(\sin t) g_1(x_1(t-\sin^2 t)) g_2(x_2(t-4\sin^2 t)) + \sin t, \end{cases} \tag{11}$$

where $g_1(u) = g_2(u) = \frac{1}{2}(|u+1| - |u-1|)$. From simple calculation, one gets:

$$\underline{b}_1 = \underline{b}_2 = M_1 = M_2 = L_1 = L_2 = 1,$$
$$\overline{c}_{11} = \overline{c}_{22} = \frac{1}{10}, \quad \overline{c}_{12} = \overline{c}_{21} = \frac{1}{16}, \quad \overline{d}_{11} = \overline{d}_{22} = \frac{1}{16}, \quad \overline{d}_{12} = \overline{d}_{21} = \frac{1}{10},$$
$$\overline{e}_{112} = \frac{1}{8}, \quad \overline{e}_{212} = \frac{1}{4}, \quad \overline{e}_{ijl} = 0,\ i,j,l = 1,2,\ ijl \neq 112,\ ijl \neq 212.$$

Then

$$-\underline{b}_1 + \sum_{j=1}^{2}(\overline{c}_{1j} + \overline{d}_{1j})L_j + \sum_{j=1}^{2}\sum_{l=1}^{2}\overline{e}_{1jl}\Big(L_jM_l + L_lM_j\Big) = -\frac{17}{40} < 0,$$
$$-\underline{b}_2 + \sum_{j=1}^{2}(\overline{c}_{2j} + \overline{d}_{2j})L_j + \sum_{j=1}^{2}\sum_{l=1}^{2}\overline{e}_{2jl}\Big(L_jM_l + L_lM_j\Big) = -\frac{7}{40} < 0,$$

which implies all the conditions in Theorem 1 are satisfied. Hence, system (11) has exactly one π-anti-periodic solution. Moreover, the π-anti-periodic solution is globally exponentially stable.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 60974132 and 10772125.

References

1. Okochi, H.: On the existence of periodic solutions to nonlinear abstract parabolic equations. J. Math. Soc. Japan 40, 541–553 (1988)
2. Chen, Y.: Anti-periodic solutions for semilinear evolution equations. J. Math. Anal. Appl. 315, 337–348 (2006)
3. Wu, R.: An anti-periodic LaSalle oscillation theorem. Appl. Math. Lett. 21, 928–933 (2008)
4. Yin, Y.: Remarks on first order differential equations with anti-periodic boundary conditions. Nonlinear Times Digest 2, 83–94 (1995)
5. Aizicovici, S., McKibben, M., Reich, S.: Anti-periodic solutions to nonmonotone evolution equations with discontinuous nonlinearities. Nonlinear Anal. 43, 233–251 (2001)
6. Chen, Y., Nieton, J.J., O'Regan, D.: Anti-periodic solutions for fully nonlinear first-order differential equations. Math. Comput. Modell. 46, 1183–1190 (2007)
7. Wang, K., Li, Y.: A note on existence of (anti-)periodic and heteroclinic solutions for a class of second-order odes. Nonlinear Anal. 70, 1711–1724 (2009)
8. Peng, G., Huang, L.: Anti-periodic solutions for shunting inhibitory cellular neural networks with continuously distributed delays. Nonlinear Anal. Real World Appl. 10, 2434–2440 (2009)
9. Shao, J.: An anti-periodic solution for a class of recurrent neural networks. J. Comput. Appl. Math. 228, 231–237 (2009)

10. Delvos, F.J., Knoche, L.: Lacunary interpolation by antiperiodic trigonometric polynomials. BIT Numer. Math. 39, 439–450 (1999)
11. Du, J., Han, H., Jin, G.: On trigonometric and paratrigonometric Hermite interpolation. J. Approx. Theory 131, 74–99 (2004)
12. Chen, H.L.: Antiperiodic wavelets. J. Comput. Math. 14, 32–39 (1996)
13. Zhang, B., Xu, S., Li, Y., Chu, Y.: On global exponential stability of high-order neural networks with time-varying delays. Phys. Lett. A 366, 69–78 (2007)
14. Lou, X., Cui, B.: Novel global stability criteria for high-order Hopfield-type neural networks with time-varying delays. J. Math. Anal. Appl. 330, 144–158 (2007)
15. Ou, C.: Anti-periodic solutions for high-order Hopfield neural networks. Comput. Math. Appl. 56, 1838–1844 (2008)
16. Huang, Z., Song, Q., Feng, C.: Multistability in networks with self-excitation and high-order synaptic connectivity. IEEE Trans. Circuits Syst. I 57, 2144–2155 (2010)
17. Hale, J.K.: Theory of Functional Differential Equations. Springer, New York (1977)

Comparisons of Single- and Multiple-Hidden-Layer Neural Networks

Takehiko Nakama

European Center for Soft Computing
Edificio Científico Tecnológico
C/Gonzalo Gutiérrez Quirós, s/n
33600 Mieres, Spain
takehiko.nakama@softcomputing.es
http://www.softcomputing.es/

Abstract. In this study we conduct fair and systematic comparisons of two types of neural networks: single- and multiple-hidden-layer networks. For fair comparisons, we ensure that the two types use the same activation and output functions and have the same numbers of nodes, feedforward connections, and parameters. The networks are trained by the gradient descent algorithm to approximate linear and quadratic functions, and we examine their convergence properties. We show that, in both linear and quadratic cases, the learning rate is more flexible for networks with a single hidden layer than for those with multiple hidden layers. We also show that single-hidden-layer networks converge faster to linear target functions compared to multiple-hidden-layer networks.

Keywords: Neural networks, comparisons, hidden layers, architectures, gradient descent learning, back propagation, function approximation, linear function, quadratic function.

1 Introduction and Preliminaries

How many hidden layers should a neural network have? This is a rather fundamental question that one may ask in designing a neural network. Surprisingly, however, very few studies have addressed this issue. Sontag [12] compared one- and two-hidden-layer networks both consisting of linear threshold units and identified a class of functions that can be properly approximated by two-hidden-layer networks but not by one-hidden-layer networks. Chester [2] presented an example of a function that can be effectively approximated by a small number of neurons when two hidden layers are used but not when only one hidden layer is used. These studies compared representational capabilities of one- and two-hidden-layer networks.

To our knowledge, no study has conducted fair and systematic comparisons of single- and multiple-hidden-layer networks. Comparisons of the two types will be considered fair if the two types of networks (1) use the same activation and output functions; (2) have the same numbers of inputs, nodes, feedforward

© Springer-Verlag Berlin Heidelberg 2011

or feedback connections, and parameters; and (3) approximate the same target function. Also the comparisons will be considered systematic if their learning or approximation capabilities are compared by methodically setting or changing learning parameters (e.g., the learning rate, the training set and its size) and the target function.

In this study we take a first step toward conducting such fair and systematic comparisons of single- and multiple-hidden-layer networks. In order to gain analytical insight, we consider simple neural networks. However, it will become clear that they exhibit fundamental differences in convergence and learning.

It is well known that, under certain regularity conditions, the approximation error between any given continuous target function and the output function of a neural network with one hidden layer can be made arbitrarily small by increasing the number of nodes (e.g., Funahashi [4], Hornik, Stinchcombe, and White [9], Hornik [8], Barron [1]). Funanashi [4] also showed that this approximation property can be achieved by a neural network with two or more hidden layers. However, we have yet to establish a theory that tells us which of the two types is preferable for a given target function when both single- and multiple-hidden-layer networks converge to the target function. Clearly it is important, both theoretically and practically, to know their convergence properties in order to create an effective and efficient neural network for a given task.

In Section 2, we construct single- and multiple-hidden-layer networks that use the same activation and output functions and have the same number of nodes, feedforward connections, parameters, and inputs. In Section 3, we describe training procedures applied to these networks. We use the gradient descent algorithm and batch training in this study. In Section 4, we train the networks to approximate linear and quadratic functions. We show that, in both linear and quadratic cases, the learning rate is more flexible for networks with a single hidden layer than for those with multiple hidden layers. We also show that single-hidden-layer networks converge faster to linear target functions compared to multiple-hidden-layer networks.

2 Construction of Single- and Multiple-Hidden-Layer Neural Networks

In this section we describe the single- and multiple-hidden-layer networks that we compare systematically. We construct simple neural networks in order to gain analytical insight and to demonstrate their fundamental differences. Let n denote the number of nodes. Consider the pair of networks with $n = 6$ shown in Figure 1. Both networks each have one input layer, which consists of $x_1, x_2, \ldots, x_5$, and one output layer, which consists of node 6. Notice that each network receives $n - 1$ inputs. Network M has five hidden layers; each of nodes 1–5 constitutes a hidden layer. Network S has only one hidden layer, which consists of nodes 1–5. However, note that the two networks have the same numbers of inputs, nodes, and feedforward connections. In order to fairly compare the single- and multiple-hidden layer networks, we must ensure that the two networks have the

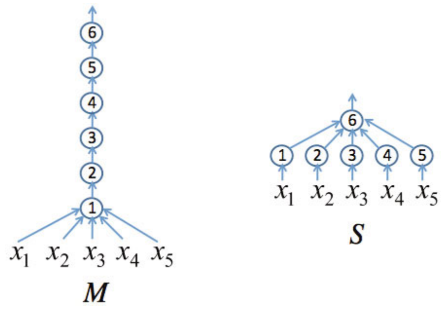


Fig. 1. Pair of networks with $n = 6$. Both networks each consist of six nodes (hence $n = 6$) and 10 feedforward connections. Network M has five hidden layers. Network S has one hidden layer.

same number of parameters and use the same activation and output functions. In fact, the two networks M and S are carefully structured so that they have the same number of parameters.

At each node, we employ a commonly used activation function: If node i receives inputs $x_1, x_2, \dots, x_m$, then we define its activation function A_i by $A_i(x_1, x_2, \dots, x_m) := \sum_{k=1}^{m} w_k^{(i)} x_k - T^{(i)}$. Let f_i denote the output function of node i. First we examine the multiple-hidden-layer network M. Let f_M denote the final output of M. Then we have

$$\begin{aligned} f_M(x_1, \dots, x_{n-1}) &= f_n(A_n(\cdots(f_1(A_1(x_1, \cdots, x_{n-1})))\cdots)) \\ &= f_n(w_1^{(n)} \cdots f_1(\textstyle\sum_{k=1}^{n-1} w_k^{(1)} x_k - T^{(1)}) \cdots - T^{(n)}). \end{aligned} \tag{1}$$

On the other hand, if we let f_S denote the final output of network S, then

$$\begin{aligned} f_S(x_1, \dots, x_{n-1}) &= f_n(A_n(f_1(A_1(x_1)), \dots, f_{n-1}(A_{n-1}(x_{n-1})))) \\ &= f_n(\textstyle\sum_{k=1}^{n-1} w_k^{(n)} f_k(w_1^{(k)} x_k - T^{(k)}) - T^{(n)}). \end{aligned} \tag{2}$$

Note that (1) and (2) have the same number of parameters. For analytical tractability, we set $T^{(k)}$ to zero for each k, and consider identity output functions: For each k, we have $f_k(t) = t$ for all t. Then we obtain

$$f_M(x_1, \dots, x_{n-1}) = (\textstyle\prod_{k=2}^{n} w_1^{(k)}) \sum_{k=1}^{n-1} w_k^{(1)} x_k. \tag{3}$$

$$f_S(x_1, \dots, x_{n-1}) = \textstyle\sum_{k=1}^{n-1} w_k^{(n)} w_1^{(k)} x_k. \tag{4}$$

It is important that f_M and f_S still have the same number of parameters.

For analytical tractability, we keep these networks simple so that we can clearly see how the two types are different mathematically. Comparing (3) and (4), we recognize that the output f_M consists of $n-1$ terms, $(\prod_{k=2}^{n} w_1^{(k)}) w_m^{(1)} x_m$, $1 \le m \le n-1$, which each include a product of n parameters. On the other hand, the output f_S consists of $n-1$ terms, $w_m^{(n)} w_1^{(m)} x_m$, $1 \le m \le n-1$, which each include a product of only two parameters. Thus, as the number of nodes increases, the number of parameters that compose the product in each term also increases for the multiple-hidden-layer network whereas it remains the same (always two) for the single-hidden-layer network. We will see that this leads to substantial differences between the two types in terms of convergence and learning.

Space limitations on this paper force us to focus on comparing the single- and multiple-hidden-layer networks for $n = 3$; we must omit our results for larger networks (we compared single- and multiple-hidden-layer networks for several values of n). However, it will become clear that the simple networks exhibit fundamental differences in convergence and learning. We found that the convergence properties observed with M and S remain qualitatively the same with larger networks. In fact, their differences are enhanced with larger networks. We will present these results in our full-length paper.

3 Network Training

We will apply the gradient descent algorithm invented by Rumelhart, Hinton, and Williams [11] to train the networks constructed in Section 2. Let $f(x_1, x_2)$ denote a target function that will be approximated by the final output function of each network, and let $\mathcal{T}$ denote the training set for M and for S. For all the networks considered in this study, we employ the same widely used quadratic loss function. The four parameters of M are $w_1^{(1)}$, $w_2^{(1)}$, $w_1^{(2)}$, and $w_1^{(3)}$ [see (3)], so we define the loss function $E_M(w_1^{(1)}, w_2^{(1)}, w_1^{(2)}, w_1^{(3)})$ for M, which will also be expressed as E_M for simplicity, by

$$\begin{aligned} E_M &:= \frac{1}{2} \sum_{(x_1,x_2)\in\mathcal{T}} [f(x_1,x_2) - f_M(x_1,x_2)]^2 \\ &= \frac{1}{2} \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - w_1^{(3)} w_1^{(2)} \textstyle\sum_{k=1}^{2} w_k^{(1)} x_k\right]^2 . \end{aligned} \tag{5}$$

There are two essential updating schemes for gradient descent learning in neural networks: batch training and on-line training (see, for instance, Fausett [3], Hassoun [5], Haykin [6], Wilson and Martinez [13]). As can be seen in (5), we use batch training in this study. The batch training scheme is a deterministic scheme that uses the exact gradient to determine the direction of each update; thus the scheme updates weights only after processing the whole epoch. The convergence properties of batch training tend to be more desirable compared to those of on-line training; see Heskes and Wiegerinck [7] and Nakama [10] for rigorous theoretical comparisons of the two updating schemes.

From (5), we obtain the following four gradients (partial derivatives) associated with M:

$$\frac{\partial E_M}{\partial w_1^{(1)}} = \textstyle\sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - w_1^{(3)} w_1^{(2)} \sum_{k=1}^{2} w_k^{(1)} x_k\right] \left(-w_1^{(3)} w_1^{(2)} x_1\right) . \tag{6}$$

$$\frac{\partial E_M}{\partial w_2^{(1)}} = \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - w_1^{(3)} w_1^{(2)} \textstyle\sum_{k=1}^{2} w_k^{(1)} x_k\right] \left(-w_1^{(3)} w_1^{(2)} x_2\right). \tag{7}$$

$$\frac{\partial E_M}{\partial w_1^{(2)}} = \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - w_1^{(3)} w_1^{(2)} \textstyle\sum_{k=1}^{2} w_k^{(1)} x_k\right] \left(-w_1^{(3)} \textstyle\sum_{k=1}^{2} w_k^{(1)} x_k\right). \tag{8}$$

$$\frac{\partial E_M}{\partial w_1^{(3)}} = \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - w_1^{(3)} w_1^{(2)} \textstyle\sum_{k=1}^{2} w_k^{(1)} x_k\right] \left(-w_1^{(2)} \textstyle\sum_{k=1}^{2} w_k^{(1)} x_k\right). \tag{9}$$

We use a constant learning rate, which will be denoted by r.

We train network S in an analogous manner. The four parameters of S are $w_1^{(1)}$, $w_1^{(2)}$, $w_1^{(3)}$, and $w_2^{(3)}$ [see (4)], so we define the loss function E_S for S by

$$E_S := \frac{1}{2} \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - \textstyle\sum_{k=1}^{2} w_k^{(3)} w_1^{(k)} x_k\right]^2. \tag{10}$$

From (10), we obtain the following four gradients associated with S:

$$\frac{\partial E_S}{\partial w_1^{(1)}} = \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - \textstyle\sum_{k=1}^{2} w_k^{(3)} w_1^{(k)} x_k\right] \left(-w_1^{(3)} x_1\right). \tag{11}$$

$$\frac{\partial E_S}{\partial w_1^{(2)}} = \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - \textstyle\sum_{k=1}^{2} w_k^{(3)} w_1^{(k)} x_k\right] \left(-w_2^{(3)} x_2\right). \tag{12}$$

$$\frac{\partial E_S}{\partial w_1^{(3)}} = \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - \textstyle\sum_{k=1}^{2} w_k^{(3)} w_1^{(k)} x_k\right] \left(-w_1^{(1)} x_1\right). \tag{13}$$

$$\frac{\partial E_S}{\partial w_2^{(3)}} = \sum_{(x_1,x_2)\in\mathcal{T}} \left[f(x_1,x_2) - \textstyle\sum_{k=1}^{2} w_k^{(3)} w_1^{(k)} x_k\right] \left(-w_1^{(2)} x_2\right). \tag{14}$$

It is informative to compare (6)–(9) and (11)–(14).

4 Comparisons of Single- and Multiple-Hidden-Layer Networks

We train the networks described in Section 2 to approximate two target functions. In Section 4.1, we approximate a linear function. In Section 4.2, we approximate a quadratic function.

4.1 Approximation of a Linear Function

We train M and S to approximate a linear target function $f_L(x_1, x_2) := x_1 + x_2$. We use $\mathcal{T} := \{(-1,-1), (-1,0), (0,-1), (0,0), (0,1), (1,0), (1,1)\}$ as the training set. In $\mathcal{T}$, the first entry of each ordered pair represents the value of x_1, and the second entry represents the value of x_2. For simplicity, the initial values of all the parameters were set to .1. (Basically the same results were obtained with other initial values.).

Figure 2 shows the approximation performances by M (the left half) and by S (the right half). Each panel plots the approximation error defined at (5) against the number of iterations, and it also indicates the value of the learning rate r. The error is shown in black if it becomes smaller than .0001 upon completion of the 1000th iteration and in light gray otherwise. Since the performances of the networks do not change noticeably after the 200th iteration, each panel shows the error only for the first 200 iterations.

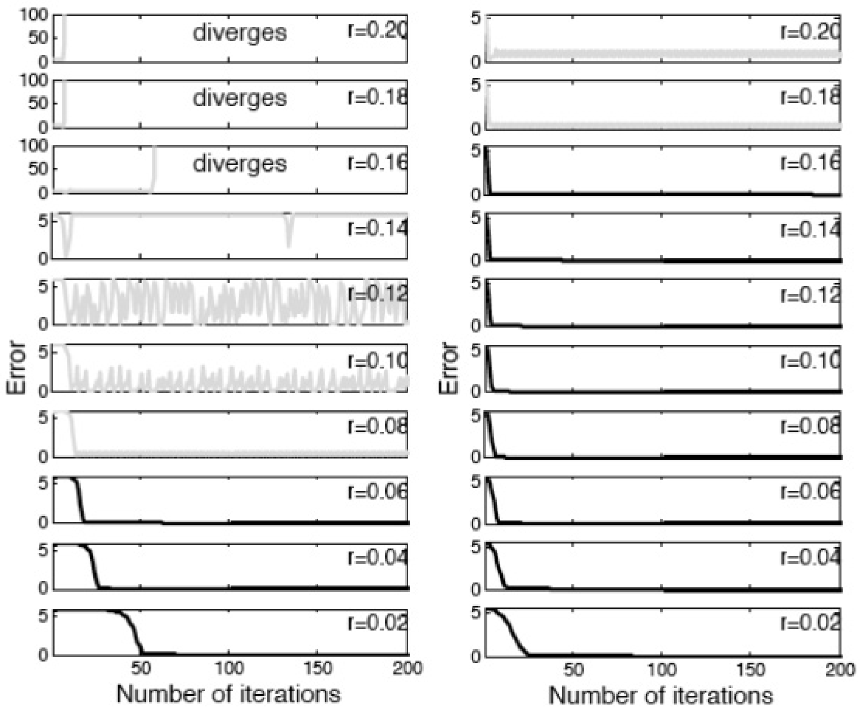


Fig. 2. Approximations of f_L by networks M (the left half) and S (the right half). Each panel plots the approximation error against the number of iterations. The error is plotted in black if it becomes smaller than .0001 upon completion of the 1000th iteration; otherwise it is plotted in light gray. In each case, the value of the learning rate r is indicated.

First we examine the approximation of f_L by M (the left half of Figure 2). The error does not converge to 0 if the learning rate r is greater than .06. When the error converges to 0, the output function $f_M(x_1, x_2) = w_1^{(3)} w_1^{(2)} \sum_{k=1}^{2} w_k^{(1)} x_k$ indeed converges pointwise to the target function. In this case, the four parameters of M converge as follows:

$$\lim_{t\to\infty} w_1^{(1)}(t) = \lim_{t\to\infty} w_2^{(1)}(t) = .7956, \ \lim_{t\to\infty} w_1^{(2)}(t) = \lim_{t\to\infty} w_1^{(3)}(t) = 1.1212.$$

Note that there are uncountably many values of the parameters with which we obtain $f_M(x_1, x_2) = x_1 + x_2$. Also, notice that the convergence slows down noticeably as the learning rate decreases from .06 to .02.

The right half of Figure 2 shows the approximation of f_L by S. First notice that the maximum value of the learning rate with which the error converges to 0 is about .16. This value is about 2.7 times as large as the corresponding value (about .06) for M. When the error converges to 0, the parameters $w_1^{(1)}$, $w_1^{(2)}$, $w_1^{(3)}$, and $w_2^{(3)}$ of the final output function $f_S(x_1, x_2) = \sum_{k=1}^{2} w_k^{(3)} w_1^{(k)} x_k$ all converge to 1 (Hence f_S converges pointwise to the target function.) Also, for all values of r with which the error converges to 0, S converges to the target function substantially faster compared to M. When $r = .00005$, the number of iterations required to make the error smaller than .0001 is 22215 for M but 14886 for S; see Figure 3. Therefore, the learning rate is more flexible for S than for M, and when the two networks converge with the same learning rate, S converges faster than M.

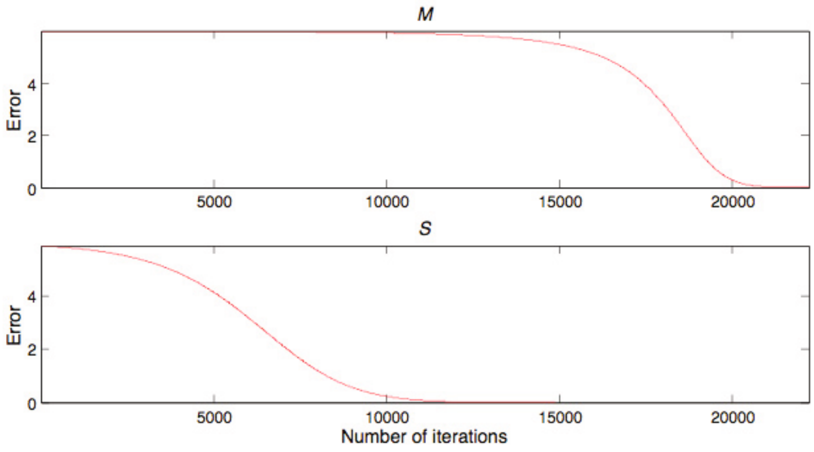


Fig. 3. Approximations of f_L with $r = .00005$

4.2 Approximation of a Quadratic Function

In this section we train M and S to approximate a quadratic target function $f_Q(x_1, x_2) := x_1^2 + x_2^2$. We again use $\mathcal{T}$ described in Section 4.1 as the training set for this approximation. Since both f_M and f_S are linear in x_1 and x_2 [see (3)–(4)], they cannot converge pointwise to f_Q. However, we can test whether the two networks converge to the least-squares linear approximation of f_Q. We denote the least-squares approximation by g and derive it. Let $g(x_1, x_2) := b_1 x_1 + b_2 x_2$ and $b := (b_1\ b_2)'$ (a' denotes the transpose of a). We will determine the entries of b that minimizes the squared error resulting from approximating f_Q by g. Let X denote a matrix that represents the training set $\mathcal{T}$. Also, let y denote a column vector whose entries are the values $f_Q(x_1, x_2)$ of the target quadratic function evaluated at the ordered pairs $(x_1, x_2) \in \mathcal{T}$: $y = (2\ 1\ 1\ 0\ 1\ 1\ 2)'$. Then b is the least-squares solution to the inconsistent system $y = Xb$, and the solution is $(X'X)^{-1}X'y = (0\ 0)'$. Thus, if the two networks properly approximate the quadratic target function, then their final output functions should converge to $g(x_1, x_2) \equiv 0$, and the least squared error in this case is

$$\frac{1}{2} \sum_{(x_1,x_2)\in\mathcal{T}} [f_Q(x_1, x_2) - g(x_1, x_2)]^2 = 6. \tag{15}$$

For simplicity, we set the initial values of all the parameters to 2.

Figure 4 shows the approximations of f_Q by M (the left half) and by S (the right half). As in each panel of Figure 2, the approximation error defined at (5) is plotted against the number of iterations, and the value of the learning rate r is indicated. The error is shown in black if it becomes smaller than 6.0001 upon completion of the 1000th iteration; in this case the output function of the network converges to the least-squares linear approximation. The error is shown in light gray otherwise (i.e., if the network does not converge to the least-squares approximation).

First we examine the approximation by M (the left half of Figure 4). The error does not converge to 6 if the learning rate is greater than .01. In each divergent case, the error increases so quickly that it is hardly visible in the corresponding plot. When the error converges to 6, the final output function $f_M(x_1, x_2) = w_1^{(3)} w_1^{(2)} \sum_{k=1}^{2} w_k^{(1)} x_k$ converges pointwise to the least-squares approximation $g(x_1, x_2) \equiv 0$; interestingly, $w_1^{(1)}(t)$ and $w_2^{(1)}(t)$ converge to 0 whereas $w_1^{(2)}(t)$ and $w_1^{(3)}(t)$ converge to -1.83.

The right half of Figure 4 shows the approximation of f_Q by S. The maximum value of the learning rate with which f_S converges to the least-squares linear solution is approximately .08, so this value is about eight times as large as the corresponding value (approximately .01) for M. When the error converges to 6, the parameters $w_1^{(1)}$, $w_1^{(2)}$, $w_1^{(3)}$, and $w_2^{(3)}$ of the final output function $f_S(x_1, x_2) = \sum_{k=1}^{2} w_k^{(3)} w_1^{(k)} x_k$ all converge to 0, so the output function converges pointwise to the least-squares solution. (However, notice that the limits of the parameters of S are different from those of M.) Therefore, as in the linear case described in Section 4.1, the effective learning rate is more flexible for S than for M.

5 Discussion

We believe that our study is the first to conduct fair and systematic comparisons of single- and multiple-hidden-layer neural networks. The two types of networks considered in this study use the same activation and output functions and have the same numbers of nodes, feedforward connections, parameters, and inputs. They were trained to approximate linear and quadratic functions. In all the cases examined in this study, the range of the effective learning rate is substantially wider for single-hidden-layer networks than for multiple-hidden-layer networks. When they converge to the linear target function f_L with the same learning rate, the single-hidden-layer network converges faster than the multiple-hidden-layer network. Therefore, at least in these cases, single-hidden-layer networks are

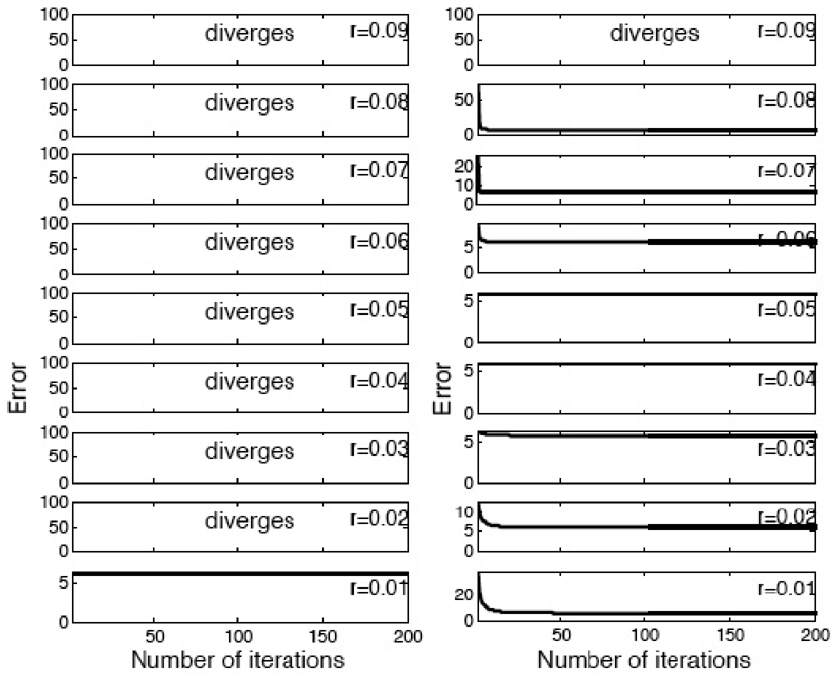


Fig. 4. Approximations of f_Q by networks M (the left half) and S (the right half). The error is plotted in black if it becomes smaller than 6.0001 upon completion of the 1000th iteration. In this case, the output function of the network converges to the least-squares linear approximation. Otherwise it is plotted in light gray. However, in each divergent case, the error increases so quickly that it is hardly visible in the corresponding panel. Note that the least squared error is 6 [see (15)].

preferable to multiple-hidden-layer networks. We examined larger networks and observed the same patterns of results; in fact, the differences between the two types are more pronounced as the network size increases.

For analytical tractability, we used identity output functions and linear and quadratic target functions in this study. We intend to compare the two types of networks with other commonly used output functions, such as sigmoidal functions, and with other types of target functions. We hope that our study serves to stimulate rigorous comparative studies of single- and multiple-hidden-layer networks.

References

1. Barron, A.R.: Approximation and estimation bounds for artificial neural networks. Machine Learning 14, 115–133 (1994)
2. Chester, D.L.: Why two hidden layers are better than one. In: Proceedings of the International Joint Conference on Neural Networks, vol. 1, pp. 265–268 (1990)

3. Fausett, L.: Fundamentals of Neural Networks. Prentice Hall, Englewood Cliffs (1994)
4. Funahashi, K.: On the approximate realization of continuous mappings by neural networks. Neural Networks 2, 183–192 (1989)
5. Hassoun, M.: Fundamentals of Artificial Neural Networks. MIT Press, Boston (1995)
6. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall, Upper Saddle River (1999)
7. Heskes, T., Wiegerinck, W.: A theoretical comparison of batch-mode, online, cyclic, and almost-cyclic learning. IEEE Transactions on Neural Networks 7, 919–925 (1996)
8. Hornik, K.: Some new results on neural network approximation. Neural Networks 6, 1069–1072 (1993)
9. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
10. Nakama, T.: Theoretical analysis of batch and on-line training for gradient descent learning in neural networks. Neurocomputing 73, 151–159 (2009)
11. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by error propagation. In: Rumelhart, D.E., McClelland, J.L., PDP Research Group (eds.) Parallel Distributed Processing, vol. 1, pp. 318–362. MIT Press, Cambridge (1986)
12. Sontag, E.D.: Feedback stabilization using two-hidden-layer nets. IEEE Transactions on Neural Networks 3, 981–990 (1992)
13. Wilson, D.R., Martinez, T.R.: The general inefficiency of batch training for gradient descent learning. Neural Networks 16, 1429–1451 (2003)

Stability of Cohen-Grossberg Neural Networks with Unbounded Time-Varying Delays

Bo Liu and Wenlian Lu

School of Mathematical Sciences, Fudan University.
220. Handan Road, Shanghai, China, 200433
liu7bo9@gmail.com,
wenlian@fudan.edu.cn

Abstract. In this paper, we consider the stability of Cohen-Grossberg neural networks with unbounded time-varying delays. Under less restrictive conditions those proposed in existing literature, we prove the asymptotic stability of the generalized Cohen-Grossberg neural networks. For the first time, we deal with the case where the unbounded time-varying delay $\tau(t)$ which is not necessarily continuous in time t. And for the amplification function, we only need it to be positive and continuous, a uniform positive lower and upper bounds are not necessary.

Keywords: stability; Cohen-Grossberg neural networks; time-varying delays.

1 Introduction

Since it was first proposed by Cohen and Grossberg in 1983 [1], the Cohen-Grossberg neural networks has been extensively studied due to its potential applications in classification and parallel computing. And there are also a lot of variants and generalization of this model. Today, the stability of Cohen-Grossberg neural networks with time delays has attracted much attention from researchers. For example, see [2, 3, 10–15] and references therein. At an early stage, the time delay is assumed as constant. After that, time-varying delays and distributed time delays are considered. In most of the papers, the time-varying delays are assume to be bounded, and only a few papers consider unbounded time delays. For example, in [11], the authors study the asymptotic stability of Cohen-Grossberg neural networks with time-varying delays:

$$\begin{aligned}\dot{x}_i(t) = -a_i(x_i(t))\Big(b_i(x_i(t)) - \sum_{j=1}^{n} c_{ij} g_j(x_j(t)) - \sum_{j=1}^{n} d_{ij} f_j(x_j(t-\tau_{ij}(t)))\Big),\\ \text{when } t \in [0,+\infty);\\ x_i(t) = \varphi_i(t) \in C([-\tau,0],\mathbb{R}),\ \text{when } t \in [-\tau,0];\\ i = 1,\cdots,n,\end{aligned}$$

where f_i, g_i are Lipschitz functions, $a_i(\cdot)$, $i = 1,\cdots,n$ are continuous and there exist positive constants m_i and M_i such that $m_i \le a_i(x) \le M_i$ for all $x \in \mathbb{R}$, and

© Springer-Verlag Berlin Heidelberg 2011

$b_i'(x) \geq \gamma_i > 0$ for all $x \in \mathbb{R}$, $\tau_{ij}(t)$ is the time-varying delay which is continuous in t, and $\tau = \sup_{x\in[0,\infty]} \max_{i,j=1,\cdots,n}(\tau_{ij}(s))$ which may be ∞. Under these assumptions, they provide sufficient conditions for the asymptotic stability of the Cohen-Grossberg networks. Yet there are two basic restrictions, one is that the time delay $\tau(t)$ is continuous in t, the other is that the amplification function $a_i(x)$ has positive lower bounds and upper bounds on $\mathbb{R}$.

In this paper, we will use a different approach which some variants of the generalized Halanay inequality to deduce the same asymptotic stability under less restrictive conditions.

The original Halanay inequality was first proved by Halanay [5]. In discussing the stability of the zero solution of

$$\dot{u}(t) = -Au(t) + Bu(t-\tau^*), \quad \tau^* > 0 \tag{1}$$

Halanay proved

Proposition 1. *(see [5, 6]) If*

$$\dot{u}(t) \leq -Au(t) + B \sup_{t-\tau^* \leq s \leq t} u(s) \tag{2}$$

and $A > B > 0$. Then, there exist k and $\gamma > 0$ such that

$$u(t) \leq ke^{-\gamma(t-t_0)} \tag{3}$$

and hence $u(t) \to 0$ when $t \to \infty$.

Later on, the original inequality was generalized in different ways to discuss the stability of solutions of Volterra functional equations of a more general type. These variants of Halanay inequality are called generalized Halanay inequality in literature. Examples of generalized Halanay inequalities include [6–9, 16].

In [7], the authors discuss stability of the following delay differential systems.

$$D^+u(t) \leq \gamma(t) + \alpha(t)u(t) + \beta(t) \cdot \sup_{t-\tau(t)\leq s \leq t} u(s) \ \ (t \geq \tilde{t}),$$

$$u(t) = |\psi(t)|, \ t \leq \tilde{t},$$

where $\psi(t)$ is a bounded and continuous function for $t \leq \tilde{t}$. For $t \in [\tilde{t}, +\infty)$, continuous functions $\gamma(t) \geq 0$, $\alpha(t) \leq 0$, $\beta(t) \geq 0$ and $\tau(t) \geq 0$, and $t - \tau(t) \to \infty$ as $t \to \infty$. They provide sufficient conditions for the boundedness and stability of the above system.

Halanay-type inequalities can be used to stability analysis of neural networks with time delays. For example, in [9], Chen proposed a approach, which is some variants of Halanay inequality, to solve global stability of the neural networks with delays:

$$\frac{du_i(t)}{dt} = -d_i u_i(t) + \sum_j a_{ij} g_j(x_j(t)) + \sum_j b_{ij} f_j(u_j(t-\tau_{ij})) + I_i, \ i = 1, \cdots, n.$$

In [16], the Halanay inequality is further generalized and used to discuss the asymptotic stability of Hopfield neural networks with unbounded time-varying delays. In this paper, motivated by the techniques used in [16], we discuss the asymptotic stability of the generalized Cohen-Grossberg neural networks with unbounded time-varying delays. We show that the asymptotic stability of the generalized Cohen-Grossberg neural networks with unbounded time-varying time delays can be proved under less restrictive conditions than those appeared in existing literature. Basically, we don't require the continuity of $\tau_{ij}(t)$, actually it can be replaced by any conditions that can ensure the existence and uniqueness of solutions. And we show that a uniform lower bound and upper bound of $a_i(x)$ on $\mathbb{R}$ is also not necessary, we only require $a_i(x)$ is positive continuous functions on $\mathbb{R}$.

The rest of the paper is organized as follows. In Section 2, we provide some mathematical preliminaries that will be used later, the main results with proof are given in Section 3, and the paper is concluded in Section 5.

2 Preliminaries

In this section, we will provide some preliminaries from matrix theory that will be used later.

Definition 1. *[4] A real $n \times n$ matrix $A = [a_{ij}]$ is said to be a non-singular M-matrix if $a_{ij} \le 0$ for $i \ne j$, $i, j = 1, \cdots, n$, and all successive principal minors of A are positive.*

Lemma 1. *[4] If $A = [a_{ij}]$ is a real $n \times n$ matrix with non-positive off-diagonal elements, then the following statements are equivalent:*

(i) *A is a non-singular M-matrix;*
(ii) *There is a positive diagonal matrix $K = \mathrm{diag}[k_1, \cdots, k_n]$, $k_i > 0$, $i = 1, \cdots, n$, such that KAK^{-1} is strictly diagonally dominant, i.e., $a_{ii} > \sum_{j \ne i} k_i |a_{ij}| k_j^{-1}$, $i = 1, \cdots, n$.*

3 Stability Analysis

In this section, we will consider the asymptotic stability of the generalized Cohen-Grossberg neural networks with time-varying delays:

$$\dot{x}_i(t) = -a_i(x_i(t))\Big(b_i(x_i(t)) - \sum_{j=1}^{n} c_{ij} g_j(x_j(t)) - \sum_{j=1}^{n} d_{ij} f_j(x_j(t - \tau_{ij}(t)))\Big), \quad i = 1, \cdots, n, \tag{4}$$

where $n \ge 2$ is the number of neurons in the network, $x_i(t)$ is the state variable of the ith neuron at time t, $a_i(x_i)$ is the amplification function, f_j and g_j are the activation functions, $C = [c_{ij}]$ is the feedback matrix and $D = [d_{ij}]$ is the delayed feedback matrix. And the initial conditions are $x_i(t) = \phi_i(t) \in C([-\infty, 0], \mathbb{R})$.

In this paper, we assume that:

Assumption 1. (i) *The activation functions $f = (f_1, \cdots, f_n)$ and $g = (g_1, \cdots, g_n)$ are Lipschitz functions, i.e., there exist positive constants α_i, β_i, $i = 1, \cdots, n$ such that $|f_i(x) - f_i(y)| \le \alpha_i |x - y|$, $|g_i(x) - g_i(y)| \le \beta_i |x - y|$;*
(ii) *$a_i(\cdot)$, $i = 1, \cdots, n$ are positive continuous functions on $\mathbb{R}$;*
(iii) *$b_i'(x) \ge \gamma_i > 0$ for all $x \in \mathbb{R}$.*

We have the following main result:

Theorem 1. *Under Assumption 1, if $t - \tau(t) \to \infty$ and $B - |CL_g| - |DL_f|$ are non-singular M-matrix, the the model (4) has a unique equilibrium point x^*, and x^* is asymptotic stable, where $\tau(t) = \max_{i,j} \tau_{ij}(t)$, $B = \mathrm{diag}[\gamma_1, \cdots, \gamma_n]$, $C = [c_{ij}]$, $D = [d_{ij}]$, $L_f = \mathrm{diag}[\alpha_1, \cdots, \alpha_n]$, $L_g = \mathrm{diag}[\beta_1, \cdots, \beta_n]$, and for a matrix $A = [a_{ij}]$, $|A| = [|a_{ij}|]$.*

Proof. *Since the existence of the equilibrium point does not depend on $a_i(\cdot)$ and $\tau(t)$, so we can use the same method as that in [11] to prove the existence of a unique equilibrium point x^*.*

Now we prove the asymptotic stability of x^*. Let $u(t)$ be a solution of the model (4), and let $v(t) = u(t) - x^*$. Since $B - |CL_g| - |DL_f|$ is a non-singular M-matrix, from Lemma 1, there exists a diagonal positive matrix $K = \mathrm{diag}[k_1, \cdots, k_m]$ such that $K(B - |CL_g| - |DL_f|)K^{-1}$ is diagonally dominant. Let $\eta > 0$ be such that

$$\gamma_i - \sum_{j=1}^{n} |c_{ij}| \beta_j k_i k_j^{-1} - \sum_{j=1}^{n} |d_{ij}| \alpha_j k_i k_j^{-1} \ge \eta, \quad i = 1, \cdots, n.$$

Let $z(t) = Kv(t)$, that is $z_i(t) = k_i v_i(t)$. For $t \ge 0$, denote $M_1(t) = \sup_{s \le t} \|z(s)\|$, where we take $\|z(t)\|$ as the infinite norm of $z(t)$, i.e., $\|z(t)\| = \max_i \|z_i(t)\|$. First, we claim that $\|z(t)\| \le M_1(0)$ for all $t \ge 0$. Otherwise, there exists t_a such that $\|z(t)\| \le M_0$ for $t \in [0, t_a)$, $\|z(t_a)\| = M_1(0)$, and $D^+\|z(t)\| \ge 0$, where D^+ is the upper-right Dini derivative which is defined as $D^+ y(t) = \overline{\lim}_{h \to 0^+} \frac{y(t+h) - y(t)}{h}$. Let i_a be the index such that $|z_{i_a}| = \|z(t_a)\|$, then

$$\begin{aligned}\left\{D^+|z_{i_a}(t)|\right\}_{t=t_a} &= -\,\mathrm{sign}(z_{i_a}(t_a)) k_{i_a} a_i(x_i(t_a)) \Big(b_{i_a}(x_{i_a}(t_a)) - b_{i_a}(x_{i_a}^*) \\ &\quad - \sum_{j=1}^{n} c_{i_a j} \left[g_j(x_j(t_a)) - g_j(x_j^*)\right] \\ &\quad - \sum_{j=1}^{n} d_{i_a j} \left[f_j(x_j(t_a)) - f_j(x_j^*)\right] \Big)\end{aligned}$$

$$\begin{aligned}
&\le -a_{i_a}(x_{i_a}(t_a))\Big(\gamma_{i_a}|z_{i_a}(t_a)| - \sum_{j=1}^{n}|c_{i_a j}|\beta_j k_{i_a}k_j^{-1}|z_j(t_a)| \\
&\quad -\sum_{j=1}^{n}|d_{ij}|k_{i_a}k_j^{-1}\alpha_j \sup_{t_a-\tau_{i_a j}(t_a)\le s\le t_a}|z_j(s)|\Big) \\
&\le -a_{i_a}(x_{i_a}(t_a))\big[\gamma_{i_a} - \sum_{j=1}^{n}|c_{i_a j}|k_{i_a}k_j^{-1}\beta_j \\
&\quad -\sum_{j=1}^{n}|d_{i_a j}|k_{i_a}k_j^{-1}\alpha_j\big]M_1(0) \\
&\le -a_{i_a}(x_{i_a}(t_a))\eta M_1(0) \\
&< 0,
\end{aligned}$$

which is a contradiction. Thus we have $\|z(t) \le M_1(0)\|$ for $t \ge 0$. Since $z(t) = K(x(t) - x^*)$, $\|x(t)\| = \|K^{-1}z(t) + x^*\| \le \frac{M_0(0)}{\min_i k_i} + \|x^*\|$. This implies that there exist constants $0 < m < M$ such that $m \le a_i(x_i(t)) \le M$ for each $i = 1, \cdots, n$ and all $t \ge 0$.

Let $\delta = \min_i \frac{\gamma_i - \sum_{j=1}^n |c_{ij}|k_i k_j^{-1}\beta_j - \frac{\eta}{2}}{\gamma_i - \sum_{j=1}^n |c_{ij}|k_i k_j^{-1}\beta_j}$, then $\delta \in (0,1)$ and

$$\delta\big(\gamma_i - \sum_{j=1}^{n}|c_{ij}|k_i k_j^{-1}\beta_j\big) - \sum_{j=1}^{n}|d_{ij}|k_i k_j^{-1}\alpha_j \ge \frac{\eta}{2}.$$

Then for any $t \ge 0$, if $\|z(t)\| \ge \delta M_1(0)$, by a similar argument as above, we can have

$$D^+\|z(t)\| \le \frac{m\eta M_1(0)}{2}.$$

Let $t_1 = \frac{2(1-\delta)}{m\eta}$, then for any $t \ge t_1$, $\|z(t)\| \le \delta M_1(0)$. Since $t - \tau(t) \to \infty$, there exists $\hat{t}_1 > t_1$ such that $t - \tau(t) \ge t_1$ for all $t \ge \hat{t}_1$. For $t \ge \hat{t}_1$, let $M_2(t) = \sup_{t_1 \le s \le t}\|z(s)\|$. Similarly, we can prove that there exists $t_2 > \hat{t}_1$ such that $\|z(t)\| \le \delta M_2(\hat{t}_1) \le \delta^2 M_1(0)$ for all $t \ge t_2, \cdots$. Continuing this process, and we can find a sequence $t_1 < t_2 < \cdots < t_k < t_{k+1} < \cdots$ such that $\|z(t)\| \le \delta^k M_1(0)$ for all $t \ge t_k$. Thus, $\lim_{t\to\infty}\|z(t)\| = 0$. The proof is completed.

4 An Example

In this section, an example is provided to illustrate the theoretical results.

Consider the following system:

$$\begin{pmatrix}\dot{x}_1\\ \dot{x}_2\end{pmatrix} = -\begin{pmatrix}e^{-x_1} & 0\\ 0 & e^{-x_2}\end{pmatrix} \times \left[\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} \times \begin{pmatrix}x_1\\ x_2\end{pmatrix} - \begin{pmatrix}\frac{1}{8} & \frac{1}{16}\\ \frac{1}{9} & \frac{1}{16}\end{pmatrix}\begin{pmatrix}\tanh x_1\\ \tanh x_2\end{pmatrix}\right.$$
$$\left. - \begin{pmatrix}\frac{1}{8} & \frac{1}{16}\\ \frac{1}{9} & \frac{1}{16}\end{pmatrix}\begin{pmatrix}\tanh 2(x_1 - \tau(t))\\ \tanh 2(x_2 - \tau(t))\end{pmatrix}\right].$$

It is easy to see that $e^x > 0$ on $\mathbb{R}$, and we can take $\gamma_i = 1$, $k_i = 1$, $\alpha_i = 2$, $\beta_i = 1$, for $i = 1, 2$. So $B = I_2$, $L_f = 2I_2$, and $L_g = I_2$, where I_2 is the identity matrix of dimension 2. We take $K = I_2$, then it is easy to verify that $B - |C| - 2|D|$ is strictly diagonally dominant. If we take $\tau(t) = \ln(t + 1)$, then it satisfies that $\lim_{t\to\infty} t - \tau(t) = \infty$, so the system is asymptotically stable from Theorem 1.

5 Conclusions

In this paper, we study the asymptotic stability of generalized Cohen-Grossberg neural networks with unbounded time varying delays. Under less restrictive conditions than existing results, we prove the asymptotic stability of such networks. Our results extended previous results concerning unbounded time varying delays in the sense that we don't require the continuity or uniform boundedness of the amplification functions. An example is provided to illustrate the theoretical results.

Acknowledgments. This work was supported by SGST 09DZ2272900 and National Science Foundation of China under Grant No.60974015.

References

1. Cohen, M.A., Grossberg, S.: Absolute Stability and Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks. IEEE Trans. Sys. Man. Cyb. 13, 815–821 (1983)
2. Arik, S., Orman, Z.: Global Stability Analysis of Cohen-Grossberg Neural Networks with Time-Varying Delays. Physics Letters A 341, 410–421 (2003)
3. Cao, J.D., Liang, J.L.: Boundness and Stability for Cohen-GrossbergNeural Networks with Time-varying Delays. J. Math. Anal. Appl. 296, 665–685 (2004)
4. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, London (1990)
5. Halanay, A.: Differential Equations. Academic Press, New York (1996)
6. Baker, C., Tang, A.: Generalized Halanay inequalities for Volterra functional differential equations and discretized versions. In: Volterra Centennial Meeting, pp. 39–55. Gordon and Breach Science Publishers, Amsterdam (1996)
7. Wen, L., Yu, Y., Wang, W.: Generalized Halanay Inequalities for Dissipativity of Volterra Functional Differential Equations. Journal of Mathematical Analysis and Applications 347, 169–178 (2008)
8. Wang, W.S.: A Generalized Halanay Inequality for Stability of Nonlinear Neutral Functional Differential Equations. Journal of Inequalities and Applications, 470519 (2010)
9. Chen, T.P.: Global Exponential Stability of Delayed Hopfield Neural Networks. Neural Networks 14, 977–980 (2001)
10. Hwang, C., Cheng, C., Liao, T.: Globally Exponential Stability of Generalized Cohen-Grossberg Neural Networks with Delays. Physics Letters A 319, 157–166 (2003)
11. Huang, T.W., Chan, A., Huang, Y., Cao, J.D.: Stability of Cohen-Grossberg Neural Networks with Time-Varying Delays. Neural Networks 20, 868–873 (2007)

12. Chen, T.P., Rong, L.B.: Delay-independent Stability Analysis of Cohen-Grossberg Neural Networks. Physics Letters A 317, 436–449 (2003)
13. Chen, T.P., Rong, L.B.: Robust Global Exponential Stability of Cohen-Grossberg Neural Networks with Time Delay. IEEE Transactions on Neural Networks 15, 203–206 (2004)
14. Rong, L.B.: LMI-Based Criteria for Robust Stability of Cohen-Grossberg Neural Networks With Delay. Physics Letters A 339, 63–73 (2005)
15. Chen, Z., Ruan, J.: Global Stability Analysis of Impulsive CohenCGrossberg Neural Networks with Delay. Physics Letters A, 101–111 (2005)
16. Liu, B., Lu, W.L., Chen, T.P.: Generalized Halanay Inequality and Their Applications to Neural Networks With Unbounded Time-Varying Delays (preprint)

Thermal Effects on Phase Response Curves and Synchronization Transition

Yasuomi D. Sato[1,2], Keiji Okumura[3], Akihisa Ichiki[3], and Masatoshi Shiino[3]

[1] Department of Brain Science and Engineering, Kyushu Institute of Technology
2-4 Hibikino, Wakamatsu-ku, Kitakyushu, 808-0196, Japan
sato-y@brain.kyutech.ac.jp

[2] Frankfurt Institute for Advanced Studies (FIAS), Johann Wolfgang Goethe University Ruth-Moufang-Str. 1, 60438,
Frankfurt am Main, Germany
sato@fias.uni-frankfurt.de

[3] Department of Physics, Faculty of Science, Tokyo Institute of Technology
2-12-1 Ohokayama, Meguro-ku, Tokyo, 152-8551, Japan
{kokumura,aichiki}@mikan.ap.titech.ac.jp,
mshiino@ap.titech.ac.jp

Abstract. We study temperature modulated synchronization phenomena in the Morris-Lecar (ML) models with synaptic couplings. Little has been known about the thermal effects on synchronization in a real nervous system. Dynamical mechanisms on such synchronization are investigated by linear stability analysis with phase descriptions for the ML type, in order to understand the effects of temperature on the phase response curve (PRC). We find two types of PRC shape modulation induced by changes in temperature that depend on an injected current amplitude: (1) the PRC shape switch between the type-I and type-II, and (2) the almost unchanged appearance of a type-II PRC. A large variety of synchronization is demonstrated with these changes in the PRC shapes.

Keywords: Temperature modulation, a phase reduction method, phase response curves, synchronization transition.

1 Introduction

Temperature is one of the most important environmental factors that influences spiking behaviors of a neuron. Changes in temperature affect the activation or inactivation of various ionic channels such as voltage-gated K^+, Na^+ and Ca^{2+} channels, which induces an action potential[1, 2]. It is well-known that temperature increases cause a constant increase in the firing frequency[3, 4] and a decrease in action potential duration[5]. This thermal dependence of the neural spiking properties has been actively studied since the electro-physiological research for model construction of Hodgkin and Huxley[6].

We are interested in studies on synchronization transitions in two coupled neurons that are influenced by temperature on their ionic or synaptic dynamics.

© Springer-Verlag Berlin Heidelberg 2011

In order to understand the thermal influence on synchronization properties of the two-neuron system, we show how a common temperature scaling factor of the ionic and synaptic dynamics affects the PRC of the neuron model, which is regarded as one of the key abilities for two coupled neurons to be synchronized.

In Sec. 2, we introduce the ML model which involves a model of the chemical synapse. The behavior of chemical synapses modeled here obeys an α-function[7, 8]. In our model, a common temperature factor, μ, is used for controlling the time constant of ionic and synaptic gating variables. We apply the phase reduction method[9–11] to the pair system of the ML model and find that temperature modulation induces two types of the PRC shape changes that are dependent on an injected current amplitude: A PRC form switch between the type-I and type-II; and an unchanged type-II PRC appearance.In Sec. 3, we explore analytically synchronous behavior in the pair system, dependent on the intrinsic parameter of a synaptic time constant. A discussion and conclusion are given in Sec. 4.

2 Spiking and Synapse Models

We begin by studying thermal effects on neuronal firing properties of the Morris-Lecar (ML) type with a physiologically plausible assumption that there exist together ionic and receptor channels on the same place of a neuron. The time constants of the channel gating variables for a potassium channel (W) and synapses (s and h) measured at the temperature $T_0 = 32$ °C (or $= 305.15$ K) are commonly scaled to ones at the temperature T by $\mu = 3.0^{(T-T_0)/10}$ [12, 13]:

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{F}(\boldsymbol{x}), \tag{1}$$

The neuronal state $\boldsymbol{x} = (V, W, s, h)^T \in \mathbb{R}^4$ and ^{T} denotes a transpose. V is a fast variable for the voltage difference across the membrane potential. The synaptic variable s is driven by the variable h, which is in turn driven by V using the step function, $E(V)$, as mentioned below. $\boldsymbol{F}(\boldsymbol{x})$ is a baseline vector field:

$$F(\boldsymbol{x}) = \begin{pmatrix} A(V,W) \\ B(V,W) \\ C(s,h) \\ D(V,h) \end{pmatrix}, \tag{2}$$

$$\begin{aligned} A(V,W) &= -\frac{1}{C_m}\left[g_L(V-V_{\mathrm{L}}) + g_{\mathrm{Ca}}m_\infty(V)(V-V_{\mathrm{Ca}})\right. \\ &\quad \left. + g_{\mathrm{K}}W(V-V_{\mathrm{K}}) - I\right], \qquad (3) \\ B(V,W) &= \frac{\mu}{\tau_w(V)}\left[W_\infty(V) - W\right], \qquad (4) \\ C(s,h) &= \frac{\mu}{\tau_{\mathrm{syn}}}(-s+h), \qquad (5) \\ D(V,h) &= \frac{\mu}{\tau_{\mathrm{syn}}}\left[-h + E(V)\right], \qquad (6) \end{aligned}$$

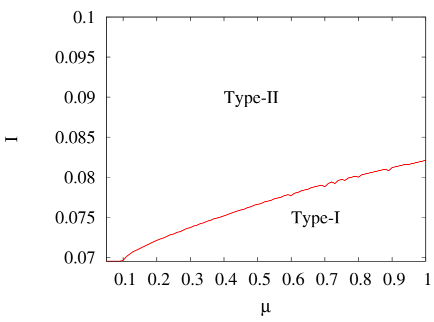


Fig. 1. An I-μ phase diagram for the PRC of the ML type. Let us define classification of the PRC type by the boundary line being determined with $C = 0.05$ in Eq. (7).

where

$$m_\infty(V) = \frac{1}{2}\left[1 + \tanh(\frac{V - V_1}{V_2})\right],$$
$$W_\infty(V) = \frac{1}{2}\left[1 + \tanh(\frac{V - V_3}{V_4})\right],$$
$$\tau_w(V) = \frac{3}{\cosh(\frac{V-V_3}{2V_4})},$$
$$E(V) = \begin{cases} 1, & x > \Theta \\ 0, & x \leqslant \Theta \end{cases},$$

where C_m is the capacitance, which takes the value $C_m = 1.00$. $g_{\mathrm{Ca}} = 1.33$, $g_{\mathrm{K}} = 2.00$, $g_{\mathrm{L}} = 0.50$, $V_{\mathrm{K}} = -0.700$, $V_{\mathrm{Ca}} = 1.000$, $V_{\mathrm{L}} = -0.500$, $V_1 = -0.010$, $V_2 = 0.150$, $V_3 = 0.100$, $V_4 = 0.145$. These dimensionless parameters were referred to [14]. Θ is a threshold with $V = 0.00$. τ_{syn} is a synaptic time constant. We assume that the injected current $I \geqslant 0.0695$ so that the ML model represents an oscillatory system exhibiting spontaneous periodic firing. Notice that when $I = 0.0695$, the ML has a phase response curve (PRC) shape of the type-I.

2.1 Temperature-Dependent PRC

we study how temperature and injected current amplitude variations influence a shape of the PRC Z_v. The PRCs are derived by a phase reduction method as written in Appendix A. Type-I PRC is well-known to take negative values as well, which are in a very minor region. So, we will have to define again the PRC classification that is different from the one that has so far been been well-known. In this work, the PRCs can be classified by measuring a difference between regional sizes of the positive and negative of the Z_v-function

$$\int_{\theta_0}^{\theta_1} Z_v(\theta')d\theta' < C\left(\int_0^{\theta_0} + \int_{\theta_1}^{1} Z_v(\theta')d\theta'\right), \tag{7}$$

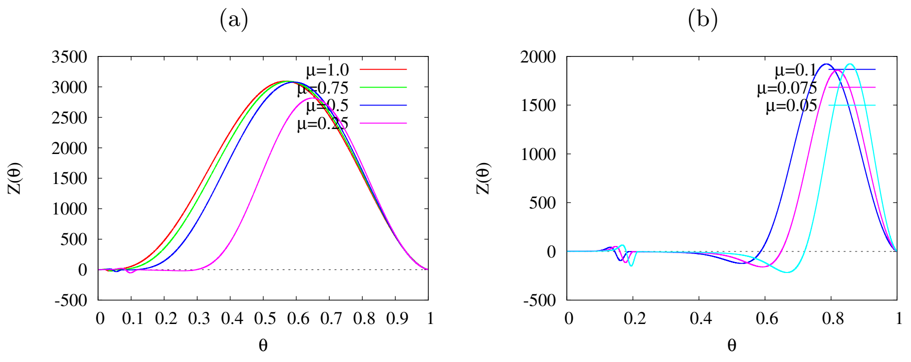


Fig. 2. μ-dependent PRCs for the ML type with $I = 0.0695$. (a) PRCs are the type-I for $\mu = 1.0$, 0.75, 0.5 and 0.25, $T_0 = 305.15$, 302.53, 298.84 and 292.53 K. (b) The type-II for $\mu(T_0) = 0.1(284.19)$, 0.75(281.57) and 0.05(277.88).

where $Z_v(\theta_0) = 0$, $Z_v(\theta_1) = 0$ and $C = 0.05$. The left hand is the size of negative region of the Z_v. The right hand presents a total size of the positive region. The PRC is called type-I if the above equation is satisfied, otherwise it is called type-II. This equation would as well describe how the timing of consequential output spike is shifted by small perturbation at an arbitrary phase. The positive and negative regions respectively correspond to the next spike advances and delays. Based on Eq.(7), we have obtained an I-μ diagram [Fig. 1]. The μ parameter region of $[0.05, 1.00]$ is set up with the temperature region of $[277.88, 305.15]$ K while the region of an injected current amplitude is $[0.0695, 0.1000]$. The region below a solid line is the type-I in our case while the other is type-II.

According to the μ-I diagram, the PRC is calculated on computers with $I = 0.0695$ in the ML model. For $\mu = 1.0$, the PRC takes a shape of the type-I which has dominantly positive values as shown in Fig. 2(a). However when the temperature parameter μ gradually decreases, the initially formed type-I PRC changes to the typical type-II PRC. It appears that the type-I PRC takes both negative and positive values (Fig. 2(b)). In the remainder of the article, we refer to this type of PRC as type-A.

Another temperature-modulated PRC is investigated for $I = 0.1000$ (Fig. 3). As shown in Fig. 3(a), we have found that the PRC can no longer take the form of type-I with any value of μ, which is unlikely for $I = 0.0695$. We refer to this type of PRC as type-B. In this case, the PRC takes only a type-II form with both the negative and positive values. However, it appears that this is an atypical form for the original type-II, as shown in FIG. 3(b).

3 Temperature Modulated Synchronization Transition in Two Neurons

Since we have found how the PRC shapes are shifted by variations of temperature and an injected current amplitude, it would be interesting to analyze the

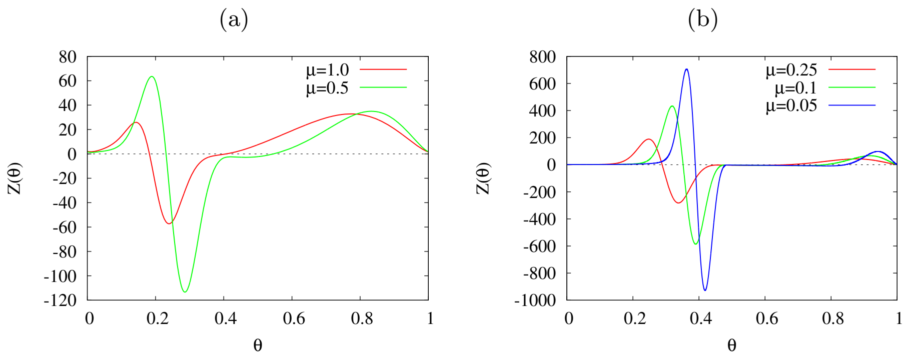


Fig. 3. μ-dependent PRCs for the ML type with $I = 0.1$. (a) $\mu = 1.0$ and 0.5. (b) $\mu = 0.25$, 0.1 and 0.05.

dynamical mechanisms of temperature modulated synchronization even in the pair system that neurons excitatorily interact with each other:

$$\frac{d\boldsymbol{x}^{(n)}}{dt} = \boldsymbol{F}^{(n)}(\boldsymbol{x}^{(n)}) + \epsilon\boldsymbol{G}^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})}), \qquad (n = 1, 2), \tag{8}$$

where $\bar{n}$ represents a counterpart of the n th neuron. $\epsilon\boldsymbol{G}^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(\bar{n})}) = (\epsilon s^{(\bar{n})}, 0, 0)^T$ where $\epsilon \ll 1$. In this analysis, we employ a phase reduction method. In the phase reduction method, the weakly coupled oscillatory models can be reduced to equations consisting of the phase degrees of freedom [Appendix A]. The obtained phase equations are given as follows:

$$\frac{d\phi}{dt} = \epsilon H_2(-\phi) - \epsilon H_1(\phi) \equiv \Gamma(\phi), \tag{9}$$

where

$$H_n(\phi) = \frac{1}{T_{\mathrm{p}}} \int_0^1 \{\tilde{\boldsymbol{Z}}^{(n)}(\theta)\}^T \cdot \boldsymbol{G}^{(n)} \left(\tilde{\boldsymbol{x}}_{\mathrm{p}}^{(n)}(\theta), \tilde{\boldsymbol{x}}_{\mathrm{p}}^{(\bar{n})}(\theta + \phi)\right) d\theta.$$

where $\phi = \theta_1 - \theta_2$ denote the phase difference between the two neurons. Γ is expressed as an average interaction function of ϕ. Z_v is a so-called PRC. Numerical calculations are used for all results. For $\Gamma(\phi)$, synchronous solutions are represented as fixed points, ϕ_0, satisfying with the condition $\Gamma(\phi_0) = 0.0$. The in-phase and anti-phase synchronous solutions are defined respectively as $\phi = 0.0$ or 1.0, and $\phi = 0.5$. In general, for $\epsilon > 0$, the synchronous solution, ϕ_0, is stable if $\Gamma'(\phi_0) < 0$, while it is unstable if $\Gamma'(\phi_0) > 0$. Since in the previous section, we explored two different types of PRC shape modulation for a change of temperature, in particular, the cases of $I = 0.0695$ and 0.1000, We will show how a temperature-scaling factor μ affects the scheme of synchronization of oscillations with respect to $\alpha = 1/\tau_{syn}$ by the stability analysis for $\Gamma(\phi)$.

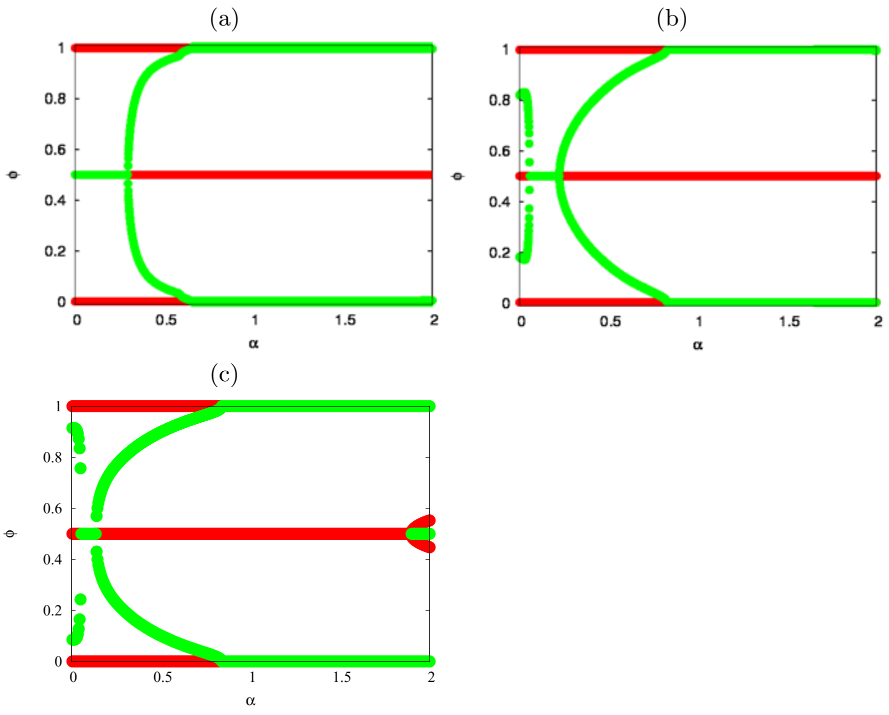


Fig. 4. μ-dependence of synchronization transition for $I = 0.0695$. (a) $\mu = 1.0$, (b) $\mu = 0.1$ and (c) $\mu = 0.05$. Light green and red lines are stable and unstable synchronous solutions, respectively.

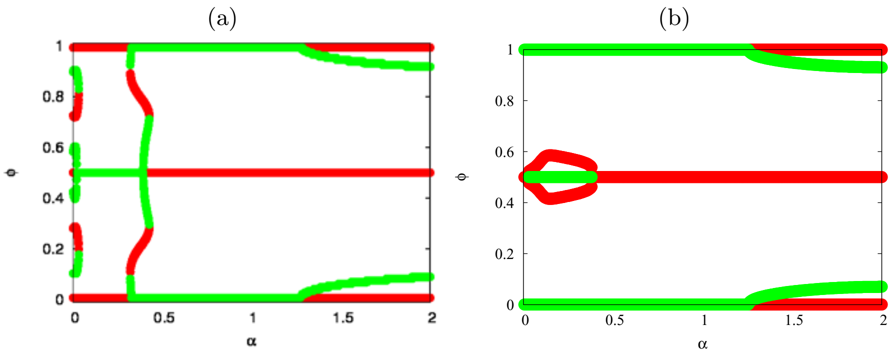


Fig. 5. μ-dependent synchronization transition in the two coupled neuronal oscillators with $I = 0.1000$. (a) $\mu = 1.0$ and (b) $\mu = 0.05$.

3.1 Type-A Synchronization Transition

We study the temperature dependence of PRC shape for $I = 0.0695$ when $\alpha = 2.0$. As shown in Fig. 2, the PRC shape is modulated for changes in μ. When μ decreases from 1.00 to 0.05, an unstable solution $\phi = 0.5$ for $\mu = 1.0$ in Fig. 4(a) splits into the two unstable synchronous solutions around the stable solution $\phi = 0.5$. Furthermore, we investigate the case for $\alpha < 2.0$. As shown in Fig. 4(a), for $\mu = 1.0$, we have found a trivial pitchfork bifurcation. This was already observed by van Vreeswijk et al. in the integrate-and-fire neuronal oscillators with α-function derived from Eqs. (5) and (6) [15, 16]. The stable anti-phase synchronous solution undergoes a transition to two stable in-phase synchronous solutions and one unstable anti-phase synchronous solution with an increase of α. However, when $\mu < 0.1$, another supercritical pitchfork bifurcation appears for small α (Fig. 4(b)). Furthermore when μ gradually decreases to 0.05, the new sub-critical pitchfork bifurcation appears as shown in Fig. 4(c).

3.2 Type-B Synchronization Transition

As in other studies on type-A synchronization, we examine simulation results for $I = 0.1000$ when $\alpha = 2.0$. In Fig. 5(a), the supercritical pitchfork bifurcation at $\alpha \sim 0.4$ has two unstable branches around $\alpha \sim 0.5$. Another supercritical pitchfork bifurcation and the two saddle-node bifurcations occur around $\alpha \sim 0.0$. When μ is decreased to 0.5, the locations of one supercritical pitchfork and two saddle-node bifurcations around $\alpha \sim 0.0$ for $I = 0.1000$ shift to slightly larger α values. The saddle-node bifurcation points for α smaller becomes close to the other for the larger α. Colliding with each other, another new unstable and stable branches appear around $\alpha \sim 0.25$ after demonstrating in Fig. 5(a). The stable branches then link to the in-phase synchronous solutions, $\phi = 0.0$ and 1.0. For $\mu = 0.05$, two sub-critical pitchfork bifurcations occur at $\phi = 0.5$ instead of two supercritical pitchfork bifurcations. One of the two unstable branches of the pitchfork bifurcation links to the other (Fig. 5(b)).

4 Discussion and Conclusion

We have investigated how synchronization in two neurons are influenced by their environmental temperature through the synaptic and ionic dynamics. Temperature modulated synchronization have not been widely reported. Using a multi-body system of globally coupled identical neurons in all-to-all fashion, Wang and Buzsáki [17] have simulated the loss in complete synchrony of the system through controlling the temperature-scaling factor. They show the system dynamically broken into two clusters; the neurons fire simultaneously within each cluster, and the spike timings of the two clusters alternate in time. However, why such phenomena occur in a decrease of temperature was not discussed. In contrast, we have explored dynamical mechanisms involved in these synchronization transitions even in the two-neuron system with the full use of phase response curve analysis, and under the physiologically reliable condition on temperature.

In this work, we have found two main types of μ-modulated PRCs. This fact implies one possibility for neuron type classification, because the ML model for $I = 0.1000$ can be regarded as type-II. Even for the HH or FHN models, type-II PRCs are found. The shapes of such PRCs do not change drastically with temperature modulation. On the other hand, it is well-known that the ML model for $I = 0.0695$ typically takes a form of the type-I. In addition, for the ML model with $I = 0.0695$, the μ-dependent PRC shape modulation between type-I and type-II has been shown. This temperature modulated PRC is expected even in the Wang and Buzsáki (WB) type model [17]. Common synchronous behavior is expected to arise in neuronal systems of the same type in our classification pair system, regardless of the detailed neuron type. When μ is small, a common dynamical property involved in synchronization phenomena is found in all models of ML, HH and WB (referred to [18]). The common dynamical property is that there are stable synchronous solutions of $\phi = 0.0$ and 1.0 for any α with sub-critical pitchfork bifurcation for $\phi = 0.5$ at an arbitrary α. However, a dynamical relationship between neuron model classification and synchronization is still open for discussion. Such a relationship should be intensively investigated in the future.

Finally, we are aware of the existence of synchronization transitions driven by the changes of α and I within a wide temperature region of $[277.88, 305.15]$ K. Our analytical approaches on synchronization phenomena were originally carried out to understand the neuronal information processing in a real brain. A human brain usually maintains a constant temperature around 37 °C. Thus, we may need to study synchronization transition in a remarkably narrow range of temperature around 37 °C under the human brain condition. The difficulty associated with this is whether extremely fast speeds of the ionic channels can be observed with small changes of temperature. A temperature coefficient and a temperature-scaling factor μ need to be considered again, according to the physiological observation of extremely fast channel speeds. So, ample discussions and open questions of our work still remain.

In summary, we have used the phase reduction method to study the thermal effects on the PRCs for the ML type. Through linear stability analysis using the phase description we obtained, temperature modulated synchronization phenomena have been found numerically in the ML type models with synaptic couplings. We have also found two different types of PRC shape changes modulated by temperature: the switch of PRC shape from the type-I to the type-II, and the mostly unchanged appearance of a type-II PRC. A variety of synchronization of oscillations have been demonstrated with these changes in the PRC shape.

Acknowledgments

Y.D.S. was supported by the Grant-in-Aid for Young Scientist (B) No. 22700237. A. I. was supported by the Grant-in-Aid for JSPS Fellows No. 20.9513. K.O. was financially supported by the Global Center of Excellence Program by MEXT, Japan through the “Nanoscience and Quantum Physics” Project of the Tokyo Institute of Technology.

References

1. Moore, J.W.: Temperature and drug effects on squid axon membrane ion conductances. Fed. Proc. 17, 113 (1958)
2. Cao, X., Oertel, D.: Temperature affects voltage-sensitive conductances differentially in octopus cells of the mammalian cochlear nucleus. J. Neurophysiol. 94, 821–832 (2005)
3. Carpenter, D.O.: Temperature effects on pacemaker generation, membrane potential, and critical firing threshold in Aplysia neurons. J. Gen. Physiol. 50(6), 1469–1484 (1967)
4. Guttman, R.: Temperature dependence of oscillation in squid axons: comparison of experiments with computations. Biophys. J. 9(3), 269–277 (1969)
5. Ishiko, N., Loewenstein, W.R.: Effects of Temperature on the Generator and Action Potentials of a Sense Organ. J. Gen. Physiol. 45(1), 105–124 (1961)
6. Hodgkin, A.L., Huxley, A.F., Katz, B.: Measurement of current-voltage relations in the membrane of the giant axon of Loligo. J. Physiol. 116, 424–448 (1952)
7. Wilson, H.R.: Spikes, Decisions, and Actions: The Dynamical Foundation of Neuroscience. Oxford University Press, New York (1999)
8. Rall, W.: Distinguishing theoretical synaptic potentials computed for different soma-dendritic distributions of synaptic input. J. Neurophysiol. 30, 1138–1168 (1967)
9. Kuramoto, Y.: Chemical Oscillations, Waves, and Turbulence. Springer, Berlin (1984)
10. Mehrotra, A., Sangiovanni-Vincentelli, A.: Noise Analysis of Radio Frequency Circuits. Kluwer Academic Publishers, Dordrecht (2004)
11. Sato, Y.D., Shiino, M.: Generalization of coupled spiking models and effects of the width of an action potential on synchronization phenomena. Phys. Rev. E 75, 011909 (2007)
12. Morris, C., Lecar, H.: Voltage oscillations in the barnacle giant muscle fiber. Biophys. J. 35(1), 193–213 (1981)
13. Wechselberger, M., Wright, C.L., Bishop, G.A., Boulant, J.A.: Ionic channels and conductance based models for hypothalamic neuronal thermosensitivity. Am. J. Physiol. Regul. Integr. Comp. Physiol. 291(3), R518–R529 (2006)
14. Ermentrout, B.: Type I membranes, phase resetting curves, and synchrony. Neural Comput. 8(5), 979–1001 (1996)
15. Van Vreeswijk, C., Abbott, L.F., Ermentrout, G.B.: Inhibition, not excitation, synchronizes coupled neurons. J. Comput. Neurosci. 1, 303–313 (1994)
16. Van Vreeswijk, C.: Partially synchronized states in networks of pulse-coupled neurons. Phys. Rev. E 54, 5522–5537 (1996)
17. Wang, X.-J., Buzsáki, G.: Gamma oscillations by synaptic inhibition in a hippocampal interneuronal network. J. Neurosci. 16(20), 6402–6413 (1996)
18. Sato, Y.D.: Synchronization Phenomena in a Pair of Coupled Neuronal Oscillator Systems. Doctoral Thesis, Tokyo Institute of Technology (2005)

A Phase Reduction Method

The general formulation of a phase reduction method is presented in [9–11]. Let $\boldsymbol{x}_{\mathrm{p}}^{(n)}(t)$ denote a stable T_{p}-periodic solution for the uncoupled dynamics,

$$\dot{\boldsymbol{x}}_{\mathrm{p}}^{(n)} = \boldsymbol{F}^{(n)}(\boldsymbol{x}_{\mathrm{p}}^{(n)}), \qquad \boldsymbol{x}_{\mathrm{p}}^{(n)}(t+T_{\mathrm{p}}) = \boldsymbol{x}_{\mathrm{p}}^{(n)}(t). \tag{10}$$

The stable solution of Eq. (1) is approximated as

$$\boldsymbol{x}^{(n)}(t) = \boldsymbol{x}_{\mathrm{p}}^{(n)}(t + \tau^{(n)}) + \epsilon \boldsymbol{u}^{(n)}(t + \tau^{(n)}(t)), \tag{11}$$

where $\tau^{(n)}(t)$ means a small perturbation in the phase direction on the periodic orbit, and $\boldsymbol{u}^{(n)}$ denotes orbital deviation from the periodic orbit, $\boldsymbol{x}_{\mathrm{p}}^{(n)}$. Substituting Eq. (11) into Eq. (8) and expanding both sides into a Taylor series, we obtain the evolution equation for $\tau^{(n)}$:

$$\begin{aligned} \dot{\tau}^{(n)}(t) = \epsilon\{\boldsymbol{Z}^{(n)}(t + \tau^{(n)}(t))\}^T \\ \cdot \boldsymbol{G}^{(n)} \left(\boldsymbol{x}_{\mathrm{p}}^{(n)}(t + \tau^{(n)}(t)), \boldsymbol{x}_{\mathrm{p}}^{(\bar{n})}(t + \tau^{(\bar{n})}(t)) \right), \end{aligned} \tag{12}$$

using the normalization condition $\{\boldsymbol{Z}^{(n)}(t)\}^T \cdot [d\boldsymbol{x}_{\mathrm{p}}^{(n)}(t)/dt] = 1$ for every t. Here $\{\boldsymbol{Z}^{(n)}(t)\}^T$ is the phase response curve, which is the unique solution to $[d\boldsymbol{Z}^{(n)}(t)/dt] = -\left[L^{(n)}(t)\right]^T \boldsymbol{Z}^{(n)}(t)$. Introducing phase variables $\theta_n(t) = (t + \tau^{(n)}(t))/T_{\mathrm{p}}$, Eq. (12) can be rewritten as

$$\frac{d\theta_n}{dt} = \frac{1}{T_{\mathrm{p}}} + \frac{\epsilon}{T_{\mathrm{p}}}\{\tilde{\boldsymbol{Z}}^{(n)}(\theta_n)\}^T \cdot \boldsymbol{G}^{(n)} \left(\tilde{\boldsymbol{x}}_{\mathrm{p}}^{(n)}(\theta_n), \tilde{\boldsymbol{x}}_{\mathrm{p}}^{(\bar{n})}(\theta_{\bar{n}}) \right). \tag{13}$$

Let the phase difference of the two oscillators $\phi(t) = \theta_2(t) - \theta_1(t)$ is a slow variable. We have obtained the evolution equation for $\phi(t)$ as

$$\frac{d\phi}{dt} = \epsilon H_2(-\phi) - \epsilon H_1(\phi) \equiv \Gamma(\phi), \tag{14}$$

where

$$H_n(\phi) = \int_0^1 d\theta \hat{H}_n(\theta, \phi).$$

Robust H_∞ Filter Design of Delayed Neural Networks

He Huang and Xiaoping Chen

School of Electronics and Information Engineering, Soochow University, Suzhou 215006, P.R. China
cshhuang@gmail.com, xpchen@suda.edu.cn

Abstract. This paper is concerned with studying the robust H_∞ filter design problem for a class of recurrent neural networks with time-varying delay. A delay-dependent criterion involving a scaling parameter is established under which the resulting filtering error system is globally asymptotically stable with a guaranteed performance in the H_∞ sense. The purpose of the introduction of the scaling parameter lies in that the developed result can be efficiently applied to the neural networks with complex dynamic behaviors, which is illustrated by an example.

Keywords: Neural networks, time-varying delay, filter design, linear matrix inequality.

1 Introduction

This paper considers the robust H_∞ filtering problem of delayed neural networks. There are two reasons why time delay should be taken into account in the neural network models. One is that time delay is frequently encountered in the electronic implementations of neural networks because of the finite switching speed of the used amplifiers. It is now well recognized that time delay is one of the main sources resulting in instability and oscillation. The other is that it would be more effective to some neural networks based applications (for examples, speed detection of moving objects and processing of moving images) when time delay is intentionally introduced. During the past few years, delayed neural networks have received increasing research interest. Among them, much effort has been devoted to the stability analysis of various neural networks with time delay. Many stability results have been reported in the literature (see, e.g., [1–5]).

On the other hand, the state estimation of large-scale delayed neural networks is also an important issue and has recently attracted lots of attention. A neural network is generally a highly interconnected network with substantial connections between neurons. As mentioned in [6], in such a large-scale neural network, it may be very difficult or even impossible to completely obtain the state information of all neurons via the available network measurements. Meanwhile, one often needs to acquire the neuron states and then utilizes them in practical applications to achieve certain objectives such as system modeling and

© Springer-Verlag Berlin Heidelberg 2011

state feedback control, etc. Therefore, it is of great importance to investigate the state estimation problem of delayed neural networks.

Recently, some interesting results on the state estimation of delayed neural networks have been published in the literature [6–13]. The authors in [6] firstly studied the state estimation of delayed neural networks and derived a delay-independent design criterion. The free-weighting matrix approach was applied to discuss this problem in [8], and a delay-dependent condition was established to ensure the existence of a suitable state estimator. The robust state estimation problem was investigated in [10, 12] for delayed neural networks where the parameter uncertainties were taken into account. Several delay-dependent results were presented in [13] to the state estimation of neutral-type neural networks with time-varying delay. It should be pointed out that only the state estimator design problem was addressed in the above literature. As suggested in [14], a neural network is often subject to noise perturbations. In some situations, the noise may be deterministic rather than random. That is, it can be modeled as an energy-bounded input noise. Therefore, the performance analysis, which was not discussed in [6–13] is another important issue from the point of view of theory and applications. Although the authors in [14] obtained some delay-independent and delay-dependent results on the robust filter design for delayed neural networks, this problem has not been fully studied yet and remains challenging. This motivates the current study.

In this paper, the robust H_∞ filtering problem is further discussed for a class of neural networks with time-varying delay, where the activation function is not required to be monotonic. A delay-dependent condition is derived under which the resulting filtering error system is globally asymptotically stable with a guaranteed performance in the H_∞ sense. It is shown that the design of a desired filter is accomplished by solving a linear matrix inequality (LMI), which can be facilitated by numerical algorithms. A scaling parameter is introduced such that the proposed result can be efficiently applied to the delayed neural networks with complex dynamic behaviors (for example, the chaotic neural networks), which can not be solved by the results in [14].

2 Problem Formulation

The delayed neural network disturbed by a noise input is represented by the following equations:

$$\begin{aligned}(N):\ \dot{x}(t) &= -Ax(t) + W_0 f(x(t)) + W_1 f(x(t-\tau(t))) \\ &\quad + J + B_1 w(t) \qquad (1)\\ y(t) &= Cx(t) + B_2 w(t) \qquad (2)\\ z(t) &= Dx(t) \qquad (3)\\ x(t) &= \phi(t) \qquad \forall t \in [-\tau, 0] \qquad (4)\end{aligned}$$

where $x(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T \in \mathbb{R}^n$ is the state vector of the neural network with n neurons; $y(t) \in \mathbb{R}^m$ is the network measurement; $z(t) \in \mathbb{R}^p$, which needs to be estimated, is a linear combination of the states; $w(t) \in \mathbb{R}^q$ is a noise input belonging to $L_2[0,\infty)$ which is the space of square integrabel vector functions over $[0,\infty)$. $A = \mathrm{diag}(a_1, a_2, \dots, a_n)$ is a diagonal matrix with $a_i > 0 (i = 1, 2, \dots, n)$. The matrices W_0 and W_1 are, respectively, the connection weight matrix and the delayed connection weight matrix. B_1, B_2, C and D are known real constant matrices with compatible dimensions. $f(x(t)) = [f_1(x_1(t)), f_2(x_2(t)), \dots, f_n(x_n(t))]^T$ is the neuron activation function. $\tau(t)$ represents a time-varying delay. $J = [J_1, J_2, \dots, J_n]^T$ is an external input vector. $\xi(t) \in \mathcal{C}([-\tau, 0]; \mathbb{R}^n)$ is a real-valued continuous initial condition on $[-\tau, 0]$.

For the delayed neural network (N), a full-order filter is designed to estimate $z(t)$:

$$\begin{aligned} (N_f): \ \dot{\hat{x}}(t) &= -A\hat{x}(t) + W_0 f(\hat{x}(t)) + W_1 f(\hat{x}(t-\tau(t))) \\ &\quad + J + K(y(t) - \hat{y}(t)) \qquad (5) \\ \hat{y}(t) &= C\hat{x}(t) \qquad (6) \\ \hat{z}(t) &= D\hat{x}(t) \qquad (7) \\ \hat{x}(0) &= 0 \qquad (8) \end{aligned}$$

where $\hat{x}(t) \in \mathbb{R}^n$, $\hat{y}(t) \in \mathbb{R}^m$, $\hat{z}(t) \in \mathbb{R}^p$, and K, to be determined, is the gain matrix of the filter.

Let the filter errors be $e(t) = x(t) - \hat{x}(t)$ and $\bar{z}(t) = z(t) - \hat{z}(t)$. The filtering error system can be immediately obtained from (N_f) and (N):

$$\begin{aligned} (\mathcal{E}): \ \dot{e}(t) &= -(A + KC)e(t) + W_0 g(t) \\ &\quad + W_1 g(t - \tau(t)) + (B_1 - KB_2)w(t) \qquad (9) \\ \bar{z}(t) &= De(t) \qquad (10) \end{aligned}$$

with

$$\begin{aligned} g(t) &= f(x(t)) - f(\hat{x}(t)), \\ g(t-\tau(t)) &= f(x(t-\tau(t))) - f(\hat{x}(t-\tau(t))). \end{aligned}$$

The problem to be addressed here is to develop an algorithm to implement an H_∞ filter design for the delayed neural network (N). Specifically, given a prescribed level of noise attenuation $\gamma > 0$, find a proper filter under which the filtering error system (9) with $w(t) = 0$ is globally asymptotically stable, and

$$\|\bar{z}\|_2 < \gamma \|w\|_2 \qquad (11)$$

under zero-initial conditions for all nonzero $w(t) \in L_2[0,\infty)$, where the norm is defined as $\|\psi\|_2 = \sqrt{\int_0^\infty \psi^T(t)\psi(t)dt}$. Then, the filtering error system $(\mathcal{E})$ is said to be globally asymptotically stable with the H_∞ performance γ.

In order to facilitate the derivative of the main result, two assumptions are always made throughout this paper:

Assumption 1. *The time-varying delay $\tau(t)$ satisfies*

$$0 \leq \tau(t) \leq \tau, \quad \dot{\tau}(t) \leq \mu \tag{12}$$

for $t > 0$, where $\tau > 0$ and μ are constant scalars.

Assumption 2. *The activation function $f(\cdot)$ satisfies:*

$$l_i^- \leq \frac{f_i(u) - f_i(v)}{u - v} \leq l_i^+, \quad u \neq v \in \mathbb{R}, \quad i = 1, 2, \ldots, n, \tag{13}$$

with l_i^- and l_i^+ being constant scalars.

Then, it follows from (13) that

$$l_i^- \leq \frac{g_i(t)}{e_i} = \frac{f_i(x_i) - f_i(\hat{x}_i)}{x_i - \hat{x}_i} \leq l_i^+, \quad e_i \neq 0, \quad i = 1, 2, \ldots, n. \tag{14}$$

Remark 1. It should be pointed out that the scalars l_i^- and l_i^+ may be positive, negative or zero [7, 9]. That is to say, it is not required that the activation function is monotonic. It is thus less restrictive than the popularly-used sigmoid functions.

Lemma 1. *[16] For any given constant matrix $\mathcal{X} \in \mathbb{R}^{m \times m}, \mathcal{X} = \mathcal{X}^T > 0$, scalar $\nu > 0$, vector function $\omega : [0, \nu] \to \mathbb{R}^m$ such that the integrations concerned are well defined, then*

$$\nu \int_0^\nu \omega^T(s) \mathcal{X} \omega(s) ds \geq \Big(\int_0^\nu \omega(s) ds \Big)^T \mathcal{X} \Big(\int_0^\nu \omega(s) ds \Big).$$

3 Main Result

This section is dedicated to presenting an approach to the H_∞ filter design problem of the neural network (N). A delay-dependent criterion is derived in terms of an LMI, which is formulated as the following theorem:

Theorem 1. *Let α be a positive constant and $\gamma > 0$ be a prescribed constant, the filtering error system $(\mathcal{E})$ is globally asymptotically stable with the H_∞ performance γ if there exist real positive definite matrices $P > 0, Q_1 > 0, Q_2 > 0, Q_3 > 0, R > 0$, two diagonal matrices $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n) > 0, \Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_n) > 0$ and a matrix G such that*

$$\begin{bmatrix} \Omega_1 & 0 & \alpha^2 R & \Omega_2 & PW_1 & \Omega_3 & \Omega_4 \\ * & \Omega_5 & \alpha^2 R & 0 & 0 & 0 & 0 \\ * & * & \Omega_6 & 0 & \Omega_7 & 0 & 0 \\ * & * & * & \Omega_8 & 0 & 0 & \alpha\tau W_0^T P \\ * & * & * & * & \Omega_9 & 0 & \alpha\tau W_1^T P \\ * & * & * & * & * & -\gamma^2 I & \Omega_{10} \\ * & * & * & * & * & * & \Omega_{11} \end{bmatrix} < 0, \tag{15}$$

where $*$ always represents the symmetric block in a symmetric matrix, and

$$\begin{aligned}
\Omega_1 &= -PA - A^TP - GC - C^TG^T + Q_1 + Q_2 - \alpha^2 R - 2L^+ \Lambda L^- + D^TD, \\
\Omega_2 &= PW_0 + L^-\Lambda + L^+\Lambda, \quad \Omega_3 = PB_1 - GB_2, \\
\Omega_4 &= -\alpha\tau A^TP - \alpha\tau C^TG^T, \quad \Omega_5 = -Q_2 - \alpha^2 R, \\
\Omega_6 &= -(1-\mu)Q_1 - 2\alpha^2 R - 2L^+\Sigma L^-, \quad \Omega_7 = L^-\Sigma + L^+\Sigma, \\
\Omega_8 &= -2\Lambda + Q_3, \quad \Omega_9 = -(1-\mu)Q_3 - 2\Sigma, \\
\Omega_{10} &= \alpha\tau B_1^TP - \alpha\tau B_2^TG^T, \quad \Omega_{11} = -2P + R, \\
L^- &= diag(l_1^-, l_2^-, \dots, l_n^-), \quad L^+ = diag(l_1^+, l_2^+, \dots, l_n^+).
\end{aligned}$$

Then, the gain matrix K of the filter is designed as

$$K = P^{-1}G.$$

Proof. Choose a Lyapunov-Krasovskii functional candidate as

$$\begin{aligned}
V(t) &= e^T(t)Pe(t) + \int_{t-\tau(t)}^{t} e^T(s)Q_1e(s)ds + \int_{t-\tau}^{t} e^T(s)Q_2e(s)ds \\
&\quad + \int_{t-\tau(t)}^{t} g^T(s)Q_3g(s)ds + \alpha^2\tau \int_{t-\tau}^{t} (s-t+\tau)\dot{e}^T(s)R\dot{e}(s)ds. \quad (16)
\end{aligned}$$

Evaluating the time-derivative of $V(t)$ along the trajectories of system (9) and noting (12) yield

$$\begin{aligned}
\dot{V}(t) \le\ & e^T(t)[-P(A+KC) - (A+KC)^TP + Q_1 + Q_2]e(t) + 2e^T(t)PW_0g(t) \\
& +2e^T(t)PW_1g(t-\tau(t)) + 2e^T(t)P(B_1 - KB_2)w(t) \\
& -(1-\mu)e^T(t-\tau(t))Q_1e(t-\tau(t)) - e^T(t-\tau)Q_2e(t-\tau) \\
& +g^T(t)Q_3g(t) - (1-\mu)g^T(t-\tau(t))Q_3g(t-\tau(t)) \\
& +\alpha^2\tau^2\dot{e}^T(t)R\dot{e}(t) - \alpha^2\tau\int_{t-\tau}^{t}\dot{e}^T(s)R\dot{e}(s)ds. \quad (17)
\end{aligned}$$

Using Lemma 1, one has

$$\begin{aligned}
-\alpha^2\tau\int_{t-\tau}^{t}\dot{e}^T(s)R\dot{e}(s)ds &= -\alpha^2\tau\int_{t-\tau}^{t-\tau(t)}\dot{e}^T(s)R\dot{e}(s)ds \\
&\quad -\alpha^2\tau\int_{t-\tau(t)}^{t}\dot{e}^T(s)R\dot{e}(s)ds \\
&\le -\alpha^2\left(\int_{t-\tau}^{t-\tau(t)}\dot{e}(s)ds\right)^T R\int_{t-\tau}^{t-\tau(t)}\dot{e}(s)ds \\
&\quad -\alpha^2\left(\int_{t-\tau(t)}^{t}\dot{e}(s)ds\right)^T R\int_{t-\tau(t)}^{t}\dot{e}(s)ds \\
&= -\alpha^2[e(t-\tau(t)) - e(t-\tau)]^TR[e(t-\tau(t)) - e(t-\tau)] \\
&\quad -\alpha^2[e(t) - e(t-\tau(t))]^TR[e(t) - e(t-\tau(t))]. \quad (18)
\end{aligned}$$

For the diagonal matrix $\varLambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n) > 0$, it follows from (14) that

$$\begin{aligned} 0 &\leq -2\sum_{i=1}^{n} \lambda_i [g_i(t) - l_i^+ e_i(t)][g_i(t) - l_i^- e_i(t)] \\ &= -2g^T(t)\varLambda g(t) + 2g^T(t)\varLambda L^- e(t) \\ &\quad +2e^T(t)L^+\varLambda g(t) - 2e^T(t)L^+\varLambda L^- e(t). \end{aligned} \tag{19}$$

Similarly, for the diagonal matrix $\varSigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n) > 0$,

$$\begin{aligned} 0 \leq &-2g^T(t-\tau(t))\varSigma g(t-\tau(t)) + 2g^T(t-\tau(t))\varSigma L^- e(t-\tau(t)) \\ &+2e^T(t-\tau(t))L^+\varSigma g(t-\tau(t)) - 2e^T(t-\tau(t))L^+\varSigma L^- e(t-\tau(t)). \end{aligned} \tag{20}$$

By combining (17)-(20) together, one can deduce that

$$\bar{z}^T(t)\bar{z}(t) - \gamma^2 w^T(t)w(t) + \dot{V}(t) \leq \zeta^T(t)[\varTheta_1 + \alpha^2\tau^2\varTheta_2^T R\varTheta_2]\zeta(t), \tag{21}$$

with

$$\begin{aligned} \zeta(t) &= [e^T(t), e^T(t-\tau), e^T(t-\tau(t)), g^T(t), g^T(t-\tau(t)), w^T(t)]^T, \\ \varTheta_1 &= \begin{bmatrix} \varPhi_1 & 0 & \alpha^2 R & \varOmega_2 & PW_1 & PB_1 - PKB_2 \\ * & \varOmega_5 & \alpha^2 R & 0 & 0 & 0 \\ * & * & \varOmega_6 & 0 & \varOmega_7 & 0 \\ * & * & * & \varOmega_8 & 0 & 0 \\ * & * & * & * & \varOmega_9 & 0 \\ * & * & * & * & * & -\gamma^2 I \end{bmatrix}, \\ \varTheta_2 &= \begin{bmatrix} -A-KC & 0 & 0 & W_0 & W_1 & B_1 - KB_2 \end{bmatrix}, \\ \varPhi_1 &= -P(A+KC) - (A+KC)^T P + Q_1 + Q_2 - \alpha^2 R - 2L^+\varLambda L^- + D^T D. \end{aligned}$$

By Schur complement, $\varTheta_1 + \alpha^2\tau^2\varTheta_2^T R\varTheta_2 < 0$ is equivalent to $\begin{bmatrix} \varTheta_1 & \alpha\tau\varTheta_2^T R \\ * & -R \end{bmatrix} < 0$. Pre- and post-multiplying it respectively by $\mathrm{diag}\{I, I, I, I, I, I, PR^{-1}\}$ and its transpose, and noting that $K = P^{-1}G$ and $-PR^{-1}P \leq -2P + R$, one can easily derive that $\varTheta_1 + \alpha^2\tau^2\varTheta_2^T R\varTheta_2 < 0$ is guaranteed by (15). It means that $\bar{z}^T(t)\bar{z}(t) - \gamma^2 w^T(t)w(t) + \dot{V}(t) < 0$ for any $\zeta(t) \neq 0$.

Define

$$J(t) = \int_0^\infty [\bar{z}^T(t)\bar{z}(t) - \gamma^2 w^T(t)w(t)]dt. \tag{22}$$

Under the zero-initial condition, it follows from (16) that $V(t)|_{t=0} = 0$ and $V(t) \geq 0$ for $t > 0$. Then, for any nonzero $w(t) \in L_2[0, \infty)$, one gets

$$\begin{aligned} J(t) &\leq \int_0^\infty [\bar{z}^T(t)\bar{z}(t) - \gamma^2 w^T(t)w(t)]dt + V(t)|_{t\to\infty} - V(t)|_{t=0} \\ &= \int_o^\infty [\bar{z}^T(t)\bar{z}(t) - \gamma^2 w^T(t)w(t) + \dot{V}(t)]dt \\ &< 0. \end{aligned} \tag{23}$$

It thus follows from (22) that $\|\bar{z}\|_2 < \gamma\|w\|_2$. On the other hand, similar to the derivative of (21), one can obtain that the filtering error system (9) with $w(t) = 0$ is globally asymptotically stable if the LMI (15) is satisfied. Due to the page limit, the procedure is omitted here. This completes the proof.

Remark 2. As can be seen from Theorem 1, a scaling parameter α is introduced. Its advantage is that Theorem 1 can be efficiently employed to tackle the H_∞ filter design problem of delayed neural networks with complex dynamic behaviors, where the results in [14] are invalid.

Remark 3. The H_∞ performance index γ described in Theorem 1 can be optimized via solving:

Algorithm 1. $\min_{P,Q_1,Q_2,Q_3,R,\Lambda,\Sigma,G} \gamma^2$ *subject to the LMI* (15).

Remark 4. It should be pointed out that the conservatism of Theorem 1 can be further reduced by using the delay decomposition approach [3, 5, 17].

4 An Illustrated Example

Let $x(t) = [x_1(t), x_2(t)]^T \in \mathbb{R}^2$. Consider a delayed neural network (N) with

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, W_0 = \begin{bmatrix} 2 & -0.1 \\ -5 & 3 \end{bmatrix}, W_1 = \begin{bmatrix} -1.5 & -0.1 \\ -0.2 & -2.5 \end{bmatrix}, J = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} 0.1 & -0.1 \end{bmatrix}^T, \quad B_2 = 0.2, \quad C = \begin{bmatrix} 0 & 1.5 \end{bmatrix}, D = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix}.$$

The activation function is given by $f(x) = \tanh(x)$ with $l_1^- = l_2^- = 0$ and $l_1^+ = l_2^+ = 1$, the time delay $\tau(t)$ is taken as $\tau(t) = 1$, and the noise disturbance is assumed to be $w(t) = \frac{1}{1+t}$. It is known from [18] that this neural network exhibits chaotic dynamic behaviors. No feasible solution can be found by Theorem 2 in [14]. However, by letting $\alpha = 0.3$ and resorting to Matlab LMI Control Toolbox, the gain matrix K of the filter can be designed as $K = \begin{bmatrix} -9.7532 \\ 15.1932 \end{bmatrix}$ with the H_∞ performance $\gamma = 0.7437$. It clearly confirms the effectiveness of the proposed result for the H_∞ filter design of delayed chaotic neural networks.

5 Conclusion

In this paper, a delay-dependent approach has been presented to deal with the robust H_∞ filter design problem for a class of neural network with time-varying delay. A design criterion has been obtained by means of an LMI, which can be easily solved by some standard software. A scaling parameter has been introduced such that the developed result can be efficiently applied to the delayed neural networks with complex dynamic behaviors including chaotic neural networks.

Acknowledgments

This work was jointly supported by the National Natural Science Foundation of China under Grant No. 61005047 and the Natural Science Foundation of Jiangsu Province of China under Grant No. BK2010214.

References

1. Ensari, T., Arik, S.: Global stability of neural networks with multiple time varying delay. IEEE Trans. Automatic Contr. 50, 1781–1785 (2005)
2. Zhang, H., Wang, Z., Liu, D.: Robust exponential stability of recurrent neural networks with multiple time-varying delays. IEEE Trans. Circuits Syst. II 54, 730–734 (2007)
3. Mou, S., Gao, H., Lam, J., Qiang, W.: A new criterion of delay-dependent asymptotic stability for Hopfield neural networks with time delay. IEEE Trans. Neural Netw. 19, 532–535 (2008)
4. Forti, M.: Convergence of a subclass of Cohen-Grossberg neural networks via the Łojasiewicz inequality. IEEE Trans. Syst. Man Cyber. B 38, 252–257 (2008)
5. Zhang, X., Han, Q.-L.: New Lyapunov-Krasovskii functionals for global asymptotic stability of delayed neural networks. IEEE Trans. Neural Netw. 20, 533–539 (2009)
6. Wang, Z., Ho, D.W.C., Liu, X.: State estimation for delayed neural networks. IEEE Trans. Neural Netw. 16, 279–284 (2005)
7. 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)
8. He, Y., Wang, Q.-G., Wu, M., Lin, C.: Delay-dependent state estimation for delayed neural networks. IEEE Trans. Neural Netw. 17, 1077–1081 (2006)
9. Huang, H., Feng, G.: A scaling parameter approach to delay-dependent state estimation of delayed neural networks. IEEE Trans. Circuits Syst. II 57, 36–40 (2010)
10. Huang, H., Feng, G., Cao, J.: Robust state estimation for uncertain neural networks with time-varying delay. IEEE Trans. Neural Netw. 19, 1329–1339 (2008)
11. Huang, H., Feng, G., Cao, J.: State estimation for static neural networks with time-varying delay. Neural Networks 23, 1202–1207 (2010)
12. Liu, X., Cao, J.: Robust state estimation for neural networks with discontinuous activations. IEEE Trans. Syst. Man Cyber. B 40, 1425–1437 (2010)
13. Park, J.H., Kwon, O.M.: State estimation for neural networks of neutral-type with interval time-varying delays. Appl. Math. Comput. 203, 217–223 (2008)
14. Huang, H., Feng, G.: Delay-dependent H_∞ and generalized H_2 filtering for delayed neural networks. IEEE Trans. Circuits Syst. I 56, 846–857 (2009)
15. Boyd, S., EI Ghaoui, L., Feron, E., Balakrishnan, V.: Linear Matrix Inequalities in System and Control Theory. SIAM, Philadelphia (1994)
16. Gu, K., Kharitonov, V.L., Chen, J.: Stability of Time-delay Systems. Birkhauser, Massachusetts (2003)
17. Gouaisbaut, F., Peaucelle, D.: Delay-dependent stability analysis of linear time delay systems. In: Proceeding of the IFAC TDS 2006, L'Aquila, Italy (2006)
18. Lu, H.: Chaotic attractors in delayed nerual networks. Phys. Lett. A 298, 109–116 (2002)

On Metastability of Cellular Neural Networks with Random Perturbations

Liqiong Zhou and Wenlian Lu

School of Mathematical Science, Fudan University,
Shanghai 200433, P.R. China
wenlian@fudan.edu.cn, 09210180050@fudan.edu.cn

Abstract. Metastability is known as the phenomenon that a multi-stable system may have transitions between different stable states under random perturbations of proper intensity. In this paper, we make a trial metastability analysis of cellular neural networks with multi-stable equilibria analytically and numerically. Via the large deviation theory, we can define the MOST stable equilibrium according to the minimum action of the transition paths between these equilibria. Under a proper intensity of white noise, the trajectories from any initial position will go and stay near the most stable equilibrium as defined, even if the trajectory is initiated right at the other stable equilibria or its attracting basins. We provide a sufficient condition to find the most stable equilibrium by estimating and comparing the minimal value of the action functional in the random perturbation theory. In addition, we give a simulation of 2-dimensional CNN system to illustrate the theoretical result.

Keywords: Cellular neural networks(CNNs), random perturbation, metastability, action functional, large deviation theory.

1 Introduction

In recently years, cellular neural network (CNN), introduced by Chua and Yang [1], [2], has been extensively investigated due to their important applications in many fields. The CNN can be written in the following standard form:

$$\dot{u}_i = -u_i + \sum_{j=1}^{n} a_{ij} v_j + I_i,\ i = 1, \cdots, n. \tag{1}$$

where

$$v_j = s(u_j),\ s(z) = \begin{cases} 1 & z > 1 \\ z & z \in [-1, 1] \\ -1 & z < -1 \end{cases}$$

is the saturation function, $u_i(t)$ represents the state of the i-th unit at time t,a_{ij} corresponds to the connection weight of the j-th unit on the i-th unit; v_j is the output of the j-th neuron; I_i stands for the external input current to the neuron i.

© Springer-Verlag Berlin Heidelberg 2011

A number of literature were engaged in studying the multi-stability of the CNN, which is thought of the key dynamical behaviors for the application to pattern formation. That is, each stable equilibrium stands for a pattern and the multi-stable dynamics form a map from the initial values to the stable equilibria, the patterns. See [3–5] and their references therein. In particular, [3] studied the attracting basins of multistable equilibria in cellular neural networks, which helps to construct this map.

What will the dynamics of (1) be if random noises occur at the right-hand side? It has been widely studied that transitions between stable states happen owing to the presence of noise in many systems arising in physics, chemistry, and biology. This is named by "metastability". In a gradient system, the transition can be measured by the potential of the system. But, how about the system without a scalar potential? [6, 7] provided an approach to study the transition path between stable attractors via the celebrated large deviation theory proposed in [11], where the action functional is proposed. Minimizing the action functional will obtain the mean time it costs from one stable state to another and the most possible path of this transition. So, it provides an index to compare the stability between equilibria. That is, if the minimal action functional value from one stable equilibrium A to another B is larger than that from B to A, we can justify that A is more stable. This implies that with a noise of a proper intensity, any trajectory initiated near or even at B will go and stay near A, according to the theory of stochastic resonance (SR) (See [8] and references therein). This phenomenon also provide a map from the system parameter to the most stable attractor if the noise is well adapted. For example, [9, 10] gave examples of logistic gate realized by SR.

In this paper, we are to identify the most stable equilibrium among all stable equilibria of the multistable CNN. Here, we accomplish this work through the theory of large deviations, by estimating and compareing the minimum action functional between the equilibria, to find the most stable one. Also, we can tune the parameters such as external input current or a_{ij} to control the transition motions between the equilibria. Then we give a two-dimensional CNN system with numerical simulation to verify our result.

2 Action Functional of CNN with Random Perturbation

We start with the general CNN as described as (1). It can be seen from [4] that we can divide R^n into 3^n sub-regions with respect to whether its components belong to $(-\infty,-1]$, $(-1,1)$, or $[1,+\infty)$. We define an index vector $\xi=[\xi_1,\cdots,\xi_n]$ with $\xi_i=-1,0,1$. So, we have 3^n such index vectors and define a set $\varXi$ comprising all such index vectors. Thus, we define the sub-region induced by ξ:

$$\varPhi_\xi=\left\{x=[x_1,\cdots,x_n]^\top\in\mathbb{R}^n:\ x_i\begin{cases}\geq 1 & \xi_i=1\\ \leq -1 & \xi_i=-1\\ (-1,1) & \xi_i=0\end{cases}\right\}.$$

If $x\in\varPhi_\xi$, then we can rewrite the CNN (1) in the following linear compact form

$$\dot{u}=(AD(\xi)-I_n)u+I(\xi),\ u\in\varPhi_\xi,\ \forall\,\xi\in\varXi,$$

with $u = [u_1, \cdots, u_n]^\top$, $D(\xi) = \mathrm{diag}[d_1(\xi), \cdots, d_n(\xi)]$, and $I(\xi) = [I_1(\xi), \cdots, I_n(\xi)]^\top$, where

$$d_i(\xi) = \begin{cases} 1 & \xi_i = 0 \\ 0 & \text{otherwise} \end{cases},\ I_i(\xi) = \sum_{j:\ \xi_j \neq 0} a_{ij}\xi_j + I_i,\ i = 1, \cdots, n.$$

The basic assumption here is to make all stable equilibria existing in the saturation sub-regions:

$$\mathbf{H}_1 : a_{ii} > 1 + \sum_{j=1}^{n} |a_{ij}| + |I_i|,\ i = 1, \cdots, n. \tag{2}$$

Then, we immediately have

Lemma 1. *Under the condition (2), the system (1) has 3^n equilibria and among them, there are 2^n stable ones; the system (1) is completely stable and the attracting basins of the stable equilibria composed of the stable manifolds of the unstable equilibria.*

Readers can refer to [3, 4] for the details of the proof.

Now, we consider the a random perturbation on the right-hand side of (1):

$$\dot{u}_i = -u_i + \sum_{j=1}^{n} a_{ij}v_j + I_i + \eta_i(t),\ i = 1, \cdots, n. \tag{3}$$

where η_i, $i = 1, \cdots, n$, are independent white noise satisfying:

$$\mathbb{E}(\eta_i) = 0,\ \mathbb{E}(\eta_i(t)\eta_i(t')) = D^2\delta(t - t'),$$

with D stands for the intensity of the noise. By the large deviation theory, the action functional works to measure the probability or the mean escape time from a stable equilibrium which can be read as an index of the stability of different attractors with respect to small random perturbation. See [11] for details. Here, we define the stability of each stable equilibrium according to the minimum action it costs to escape from its attracting basin. Let u^ξ be a stable equilibrium of (1) lying in Φ_ξ, $B(u^\xi)$ be its attracting basin, and $\partial B(u^\xi)$ be the boundary of $B(u^\xi)$, of which the structure is well analyzed in our previous paper [3]. Thus, we define the following functional:

$$\inf_{x(t)\in C^1([0,+\infty),\mathbb{R}^2)} \int_0^T \|\dot{x}(t) - F(x(t))\|_2^2 dt$$
$$\text{s.t. } x(0) = u^\xi \text{ and } x(T) \in \partial B(u^\xi) \text{ for some } T > 0, \tag{4}$$

where $F(x)$ represents the $\mathbb{R}^2 \to \mathbb{R}^2$ map of the right-hand side of (1). We define the value of the infimum above as an index χ_ξ to measure the stability of u^ξ in the sub-region Φ_ξ. The mean escape time τ and the probability p can be expressed as the following large deviation form as $D \to 0$ [11]:

$$\tau \sim \exp\left(\frac{\chi_\xi}{2D^2}\right),\ p \sim \exp(-\frac{\chi_\xi}{2D^2}).$$

Definition 1. *In the system (1), one stable equilibrium u^{ξ} is said to be more stable than another stable equilibrium $u^{\xi'}$ if $\chi_{\xi} > \chi_{\xi'}$.*

Therefore, the most stable equilibrium is defined as $\xi^* = \arg\max_{\xi} \chi_{\xi}$.

We carry out the variational calculus to search the necessary and sufficient condition for locally minimum of the following problem: for a given $T > 0$ and $x(T) = u^1 \in \partial B(u^{\xi})$,

$$\min_{x(t)\in C^1([0,T],\mathbb{R}^2)} \int_0^T \|\dot{x}(t) - F(x(t))\|_2^2 dt$$
$$\text{s.t. } x(0) = u^* \text{ and } x(T) = u^1. \tag{5}$$

It is clear that if we relax the $T > 0$ and $u^1 \in \partial B(u^*)$, then we can obtain the infimum of (4).

First, we take the case of $n = 1$ of (3) to illustrate the main idea. We consider the following system:

$$\dot{u} = -u + av + I + \eta(t). \tag{6}$$

So the potential of this system is

$$H = \begin{cases} \frac{1}{2}u^2 + au - Iu, u < -1 \\ \frac{1}{2}u^2 - \frac{a}{2}u^2 - Iu, -1 \leq u \leq 1 \\ \frac{1}{2}u^2 - au - Iu, u > 1. \end{cases}$$

It's been proved that when a and I are in the proper range, there are three equilibria and two of them are stable and the other is unstable. For $u < -1$, then the corresponding equilibrium is $u^1 = -a + I$, which is clearly stable. In this case, the potential is $H_1 = \frac{1}{2}u^2 + au - Iu$. So the increasing potential it needs to arrive at the edge $\xi = -1$ is $\varDelta H_1 = (\frac{1}{2} - a + I) - (\frac{1}{2}(-a+I)^2 + a(-a+I) - I(-a+I)) = \frac{1}{2}[1-(a-I)]^2$.

For $-1 \leq u \leq 1$, the equilibrium u^2 satisfies $u^2 = \frac{I}{1-a}$, which can be seen unstable. For $u > 1$, via the same process above, we get the equilibrium u^3 satisfying $u^3 = a+I$, which is also stable.

After the trajectory passes the edge $\xi = -1$, it will approach u^2. Now its potential becomes $H_2 = \frac{1}{2}u^2 - \frac{a}{2}u^2 - Iu$. So the increasing potential it needs to approach u^2 is $\varDelta H_2 = [\frac{1}{2}(\frac{I}{1-a})^2 - \frac{a}{2}(\frac{I}{1-a})^2 - I\frac{I}{1-a}] - (\frac{1}{2} - \frac{a}{2} + I) = \frac{1}{2(a-1)}[1-(a-I)]^2$. Use the same method, we can get the increasing potential which the trajectory costs to arrive at edge $\xi = 1$ from u^3 is $\varDelta H_3 = \frac{1}{2}[1-(a+I)]^2$. Passing the edge of $\xi = 1$, its potential cost to approach u^2 is $\varDelta H_4 = \frac{1}{2(a-1)}[1-a-I]^2$. Therefore, in order to compare the stability of these two stable equilibria, we just need to compare the following quantities:

$$\begin{cases} \text{If } \varDelta H_1 > \varDelta H_3 + \varDelta H_4, \text{then } u^1 \text{ is the more stable one.} \\ \text{If } \varDelta H_3 > \varDelta H_1 + \varDelta H_2, \text{then } u^3 \text{ is the more stable one.} \end{cases}$$

For the general case, according to the large deviation theory and the definition 1 above, we can give an estimating means to compare the stability of different stable equilibria of the system (3).

Theorem 1. *If there is an index vector ξ with all components nonzero such that*

$$\min_i \frac{1}{2}[I_i(\xi)-\xi_i]^2 > \max_{\substack{\zeta\neq\xi,\\ \delta(\zeta)=0}} \min_{\zeta'\in\mathcal{N}(\zeta)} \frac{1}{2}\Bigg\{[I_{i_{\zeta'}}(\zeta)-\zeta_{i_{\zeta'}}]^2 + \frac{[(a_{i_{\zeta'}i_{\zeta'}}-1)\zeta_{i_{\zeta'}}+I_{i_{\zeta'}}(\zeta')]^2}{a_{i_{\zeta'}i_{\zeta'}}-1}\Bigg\},$$

then χ_ξ is the largest one over all $\{\zeta:\ \delta(\zeta)=0\}$.

This implies that the stable equilibrium u^ξ is the most stable equilibrium over all stable ones. The proof of the Theorem 1 can be presented via two propositions as follows.

Proposition 1. *For each ξ with all components nonzero, it holds*

$$\chi_\xi \geq \min_i \frac{1}{2}[I_i(\xi)-\xi_i]^2$$

Proof. First, we consider an arbitrary stable equilibrium $u^\xi \in \Phi_\xi$ with some ξ satisfying $\xi_i \neq 0$ for all $i=1,\cdots,n$. The differential equation in this sub-region becomes:

$$\dot{u}_i = -u_i + I_i(\xi),\ i=1,\cdots,n. \tag{7}$$

The equilibrium u^ξ satisfies:

$$u_i^\xi = I_i(\xi),\ i=1,\cdots,n.$$

Thus, we have its potential as

$$H_\xi(u) = \sum_{i=1}^{n}\left[\frac{1}{2}u_i^2 - I_i(\xi)u_i\right],$$

which satisfies

$$\dot{u} = -\partial_u H_\xi(u),\ u\in\Phi_\xi. \tag{8}$$

Since if a trajectory reach $\partial B(\xi)$ from a small neighborhood of u^ξ, then it must pass the edges of the sub-region Φ_ξ, we immediately can get the proposition after computation.

Proposition 2. *For each ξ with all components nonzero, it holds*

$$\chi_\xi \leq \min_{\xi'\in\mathcal{N}_\xi} \frac{1}{2}\Bigg\{[I_{i_{\xi'}}(\xi)-\xi_{i_{\xi'}}]^2 + \frac{[(a_{i_{\xi'}i_{\xi'}}-1)\xi_{i_{\xi'}}+I_{i_{\xi'}}(\xi')]^2}{a_{i_{\xi'}i_{\xi'}}-1}\Bigg\}$$

where $i_{\xi'}$ stands for the index distinguishing ξ and its neighbor ξ'.

Proof. After proofing the proposition 1, we continue it letting i_0 be the index of the minimum value of the above, the focus point where the minimum trajectory leaves Φ_ξ is $u^3=[u_1^3,\cdots,u_n^3]^\top$ with

$$u_i^3 = \begin{cases} u_i^\xi & i\neq i_0 \\ \xi_{i_0} & i=i_0. \end{cases}$$

Since it is not a gradient system when $u(t) \in \Phi_{\xi'}$ for any ξ' with some components zeros but the other nonzero, we can not find a potential to calculate the action functional minimum. Here, we just concentrate on one trajectory which reaches the boundary of the attracting basin and is a locally minimum one. Let ξ' be an index of a neighbor sub-region of Φ_ξ with one component of ξ' zero whose index is i_0. Without loss of generality, we suppose $\xi'_1 = 0$ and the other same with ξ and consider a trajectory from u^3 to $u^{\xi'}$ by the following equations:

$$\dot{u}_1 = u_1(1 - a_{11}) - I_1(\xi') \tag{9}$$

$$\dot{u}_i = -u_i + a_{i1}u_1 + I_i(\xi'),\ i \geq 2, \tag{10}$$

with $u(0) = u^3$. It can be seen that $\lim_{t\to\infty} u(t) = u^{\xi'}$. Noting that $\dot{u}_i = F_i(u)$ for all $i \geq 2$, the only contribution to the action functional comes from the first equation, which is a gradient system with the following potential:

$$H_{\xi'}(u) = \frac{1}{2}(a_{11} - 1)u_1^2 + I_1(\xi')u_1.$$

So with the proof of proposition 1, we can get the result of the proposition 2 after computation.

Therefore, by the proposition 1 and 2, Theorem 1 is derived as a direct consequence from Propositions 1 and 2. It indicates that the stable equilibrium u^ξ is the most stable one if the index χ_ξ is the largest one over all $\{\zeta :\ \delta(\zeta) = 0\}$.

3 Numerical Illustrations: Two-Dimensional CNN System

We consider a two-dimensional CNN system as follows:

$$\begin{cases} \dot{u}_1 = -u_1 + a_{11}v_1 + a_{12}v_2 + I_1 + \eta_1(t) \\ \dot{u}_2 = -u_2 + a_{21}v_1 + a_{22}v_2 + I_2 + \eta_2(t). \end{cases} \tag{11}$$

Here, we pick $a_{11} = 3.5$, $a_{12} = 1$,$a_{21} = 0.5$, $a_{22} = 3$, $\eta_{1,2}(t)$ are independent noises with variance D^2, determined below. From Lemma 1, it can easily be verified that the system without noise has 8 equilibria in the sub-regions and 4 stable equilibria in the saturation sub-regions.

First, let $(I_1, I_2) = (-1, -1)$. Then we can get 4 stable equilibria lying in the 4 sub-regions respectively as follows:

$$u^\xi = \begin{cases} (3.5, 2.5) & \xi = (1, 1) \\ (-3.5, 1.5) & \xi = (-1, 1) \\ (1.5, -3.5) & (1, -1) \\ (-5.5, -4.5) & \xi = (-1, -1) \end{cases},$$

According to the Theorem 1, the stable equilibrium corresponding to $(-1, -1)$ is the most stable one. In fact, to the inequality (7), the minimal of its left is 12.25, and the max of its right is 9.375. So the stable equilibria in other saturation sub-regions all transit to

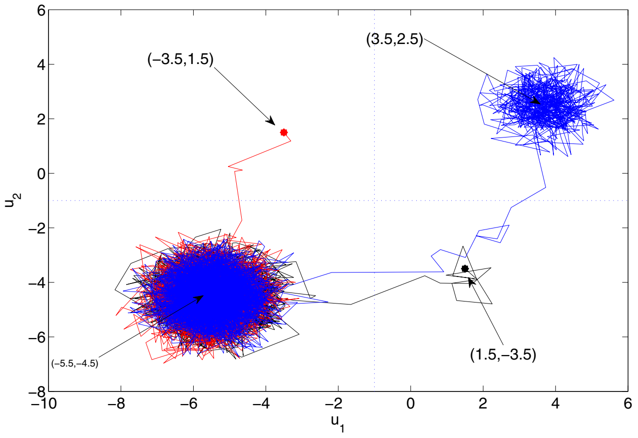


Fig. 1. The phase transition from other stable equilibria to the most stable one with $D = 1$. The trajectories are plotted by initial position at other stable equilibria: the red for the transition from the equilibrium corresponding to $(-1, 1)$, the black for $(1, -1)$, and the blue for $(1, 1)$.

the stable one in region $\Phi_\xi = (-1, -1)$ with a properly selected noise intensity $D = 1$, which is as Fig. 1 shows.

With a similar reasoning, if $(I_1, I_2) = (1, 1)$, we can know the stable equilibrium corresponding to $(1, 1)$ is the most stable one. The same result can be suggested by computing the (7), which tells the minimal of the left is 12.25, the maximal of right is 9.375, too. Therefore, the result of Fig. 2 is in accordance with the Theorem 1.

In order to show the result more sufficiently, Fig. 3 tells that the output transition from other stable equilibria to the most stable one, namely $v(t)$. During the time interval $[0, 8000]$, the output switches between $v = (1, 1)$ corresponds to $(I_1, I_2) = (1, 1)$ and $v = (-1, -1)$ corresponds to $(I_1, I_2) = (-1, -1)$.

Also, we consider the probability of succeeding transitions from other stable equilibria to the most stable one when $(I_1, I_2) = (1, 1)$ and $(I_1, I_2) = (-1, -1)$ respectively. Because not all variance D^2 is suitable and the desired output is obtained only for optimal noise intensities. Here, P is defined as the probability of $u(t)$ stays in the sub-region Φ_ξ, in other words, $v(t) = \xi$, where ξ corresponds to the most stable equilibrium, with initial values at another stable equilibrium. We calculate the probability by counting how long $v(t) = \xi$ in a long time interval $[0, T]$:

$$P = \frac{\mu\Big(\{t \in [0, T] : \ v(t) = \xi\}\Big)}{T},$$

where $\mu(\cdot)$ denotes the Lebesgue measure in $\mathbb{R}$. As shown by Figs. 4 and 5, we can explore that when D varies in a proper interval, for example between 0.3 to 1.6 approximately in our examples, the probability is equal to about 1. It means that the SR can be realized.

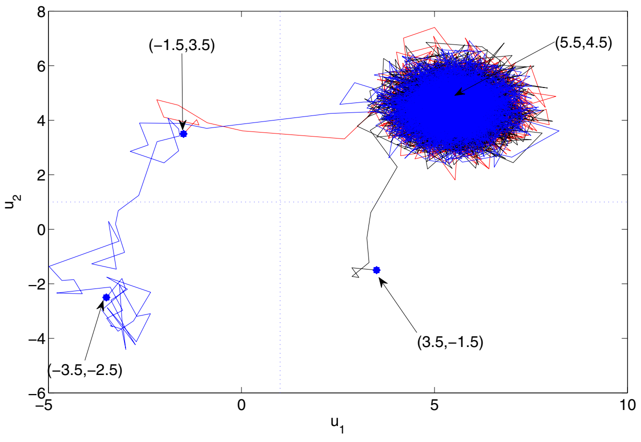


Fig. 2. The phase transition from other stable equilibria to the most stable one with $D = 1$. The trajectories are plotted by initial position at other stable equilibria: the red for the transition from the equilibrium corresponding to $(-1, 1)$, black for $(1, -1)$, and blue for $(-1, -1)$.

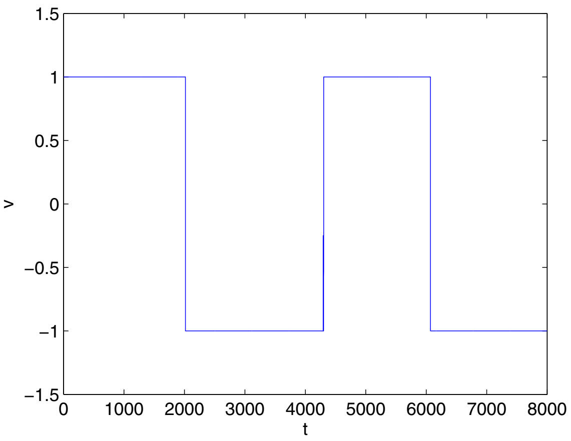


Fig. 3. The output of the system corresponding to different input(I_1, I_2) in different time intervals with $D = 1$

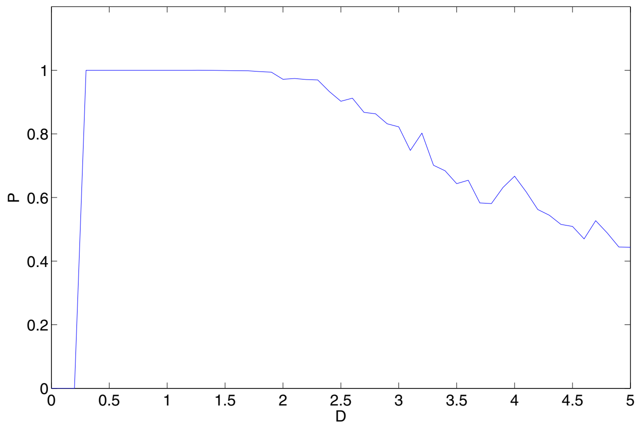


Fig. 4. The probability of succeeding transition from other stable equilibria to the most stable one corresponding to $(1, 1)$ with different D. We pick $a_{11} = 3.5$, $a_{12} = 1$, $a_{21} = 0.5$, $a_{22} = 3$.

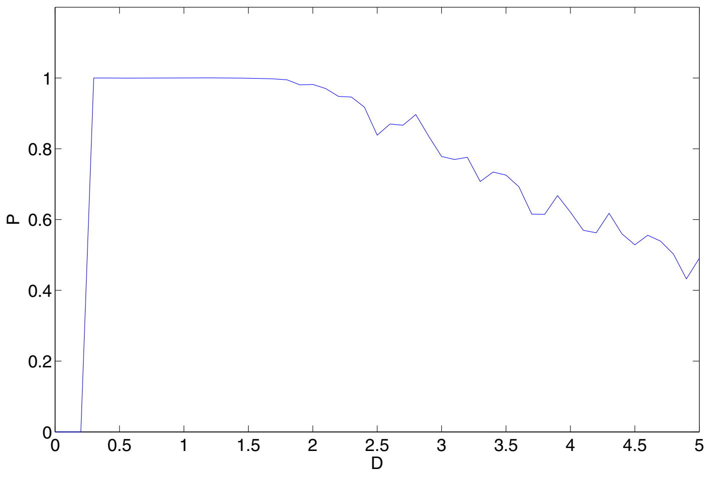


Fig. 5. The probability of succeeding transition from other stable equilibria to the most stable one corresponding to $(-1, -1)$ with different D. We pick $a_{11} = 3.5$, $a_{12} = 1$, $a_{21} = 0.5$, $a_{22} = 3$.

4 Conclusions

In this paper, we provide an approach to analyze the metastability of CNN systems based on the large deviation theory [11], via estimating the minimal action functional of the transitions between stable equilibria. Then we compare the two results and gain the Theorem 1, by an example of two-dimensional CNN system, where we give the numerical simulation to testify Theorem 1. This paper is only an initial of this topic and many issues need further exploration. These will be our research concentration in the future.

Acknowledgments

This work was supported by the National Science Foundation of China under Grant No.60974015 the Foundation for the Author of National Excellent Doctoral Dissertation of China No. 200921, SGST 09DZ2272900, and LMNS, Fudan University.

References

1. Chua, L.O., Yang, L.: Cellular neural networks:theory. IEEE T. Circuits Syst. 35(10), 1257–1272 (1988)
2. Chua, L.O., Yang, L.: Cellular neural networks:applications. IEEE T. Circuits Syst. 35(10), 1273–1290 (1988)
3. Lu, W.L., Wang, L., Chen, T.: On Attracting Basins of Multiple Equilibria of Cellular Neural Networks. IEEE Trans. Neural Netw. (2011) (accepted)
4. Wang, L., Lu, W.L., Chen, T.: Coexistence and Local Stability of Multiple Equilibria in Neural Networks with Piecewise Linear Nondecreasing Activation Functions. Neural Netw. 23, 189–200 (2010)
5. Cheng, C., Lin, K., Shih, C.: Multistability in Recurrent Neural Networks. SIAM Journal on Applied Mathematics 66(4), 1301–1320 (2006)
6. Weinan, E., Ren, W., Vanden-Eijnden, E.: Minimum Action Method for the Study of Rare Events. Communications on Pure and Applied Mathematics LVII, 0637–0656 (2004)
7. Ren, W., Vanden-Eijndenb, E., Maragakisc, P., Weinan, E.: Transition pathways in complex systems: Application of the finite-temperature string method to the alanine dipeptide. Journal of Chemical Physics 123, 134109 (2005)
8. Gammaitoni, L., Hänggi, P., Jung, P., Marchesoni, F.: Stochastic resonance. Reviews of Modern Physics 70(1), 223–287 (1998)
9. Murali, K., Rajamohamed, I., Sinha, S., Ditto, W.L., Bulsara, A.R.: Realization of reliable and flexible logic gates using noisy nonlinear circuits. Applied Physics Letters 95, 194102 (2009)
10. Murali, K., Sinha, S., Ditto, W.L., Bulsara, A.R.: Reliable Logic Circuit Elements that Exploit Nonlinearity in the Presence of a Noise Floor. Physical Review Letters 102, 104101 (2009)
11. Freidlin, M.I., Wentzell, A.D.: Random Perturbations of Dynamical Systems. Springer, Heidelberg (1984)

Fractional-Order Boundary Controller of the Anti-stable Vibration Systems

Yanzhu Zhang, Xiaoyan Wang, and Yanmei Wang

College of Information Science and Engineering,
Shenyang ligong University,
Shenyang, China
syzd710471@sina.com

Abstract. This paper discusses the boundary controller of string vibration systems, a new boundary controller of string anti-stable vibration system based on fractional calculus will be considered. This paper presents a boundary control method of anti-stable vibration systems. The fractional controller of the anti-stable system is obtained from the integer-order controller by replacing the first-order time derivative with a fractional derivative of order b. Numerical simulations are used to illustrate the improvements of the proposed controller for the anti-stable vibration systems.

Keywords: fractional calculus, adaptive control, boundary control.

1 Introduction

Boundary control of the diffusion-wave equation has been studied extensively [1][2]. In this paper, it is mainly discussed that the boundary controllers of string vibration systems. A new boundary control method based on fractional calculus of string vibration system will be considered. Fractional calculus is the field of mathematical analysis, which deals with the investigation and applications of integrals and derivatives of arbitrary order, which can be real or complex derivatives and integrals to arbitrary orders are referred to as differ integrals by many authors. It starts to play an important role in many branches of science during the last three decades. The fractional controller is obtained from the standard controller by replacing the first-order time derivative with a fractional derivative of order b, It mainly discusses the in the using fractional calculus tool.

This paper presents a boundary control method of anti-stable vibration systems. The paper is organized as follows. The first problem is boundary control of the integer-order diffusion equation, and we give the mathematical description of the boundary control of the integer-order diffusion equation, sees Section 2. The second problem is fractional-order boundary control of the anti-stable diffusion equation, and we give thesees Section 3. The basic principle of the introduced method is simple. No extra software is needed except Matlab and the Matlab Symbolic Math Toolbox. Numerical simulations are used to illustrate the improvements of the proposed control method for the anti-stable vibration systems.

© Springer-Verlag Berlin Heidelberg 2011

2 Problem Definition

We consider the vibration system governed by diffusion equation:

$$\begin{aligned} &u_t(x,t) = u_{xx}(x,t) \\ &u_x(0,t) = -Au_t(0,t) \\ &u_x(1,t) = f(t) \end{aligned} \tag{1}$$

where $f(t)$ is the control input and A is a constant parameter. For $A = 0$, equations (1) model a string which is free at the end $x = 0$ and is actuated on the opposite end. For $A < 0$, the system (1) model a string which is fixed at one end, and stabilized by a boundary controller $u = -ku(1,t), k > 0$ at the other end. For $A > 0$, the system (1) model a string which is fixed at one end, and the free end of the string is negatively damped, so that all eigen values located on the right hand side of the complex plane, so the open-loop plant of this kind of string system is "anti-stable".

In this paper we study the string system with $A > 0$, bescouse the vibration system is anti-stable system, so we want to transfer the anti-stable plant to the stable system. We map the equation (1) into the following target system. As will be shown later, the transformation (4) is invertible in a certain norm, so that stability of the target system ensures stability of the closed loop system:

$$\begin{aligned} &v_t(x,t) = v_{xx}(x,t) \\ &v_x(0,t) = bv_t(0,t) \\ &v_x(1,t) = f_d(t) \end{aligned} \tag{2}$$

which is exponentially stable for $B > 0$.

Consider the follow transformation for the anti-stable wave equation:

$$v(x,t) = u(x,t) - \int_0^x m(x,y)u_t(y,t)dy - \int_0^x n(x,y)u_x(y,t)dy) \tag{3}$$

where the gains $\boldsymbol{m(x,y), n(x,y)}$ are to be determined.

Substituting (3) into (2) we obtain:

$$\begin{aligned} v_t(x,t) &= u_t(x,t) - (\int_0^x m(x,y)u_t(y,t)dy \\ &\quad - \int_0^x n(x,y)u_x(y,t)dy)_t \\ &= u_t(x,t) - (A(x,t) - B(x,t))_t \end{aligned} \tag{4}$$

$$A(x,t) = \int_0^x m(x,y)u_t(y,t)dy$$

$$B(x,t) = \int_0^x n(x,y)u_x(y,t)dy$$

$$v_{xx}(x,t) = u_{xx}(x,t) - (A(x,t) - B(x,t))_{xx} \tag{5}$$

From (4) (5), we obtain:

$$\begin{aligned}(v(x,t))_t &= v_{xx}(x,t)) + (A(x,t) - B(x,t))_{xx} \\ &\quad -(A(x,t) - B(x,t))_t \\ &= v_{xx}(x,t)) + (A_{xx}(x,t) - (A(x,t))_t) \\ &\quad +((B(x,t))_t - B_{xx}(x,t))\end{aligned} \tag{6}$$

Matching all the terms, we get two equations as follow:

$$\begin{aligned}A_{xx}(x,y) &= (A(x,y))_t \\ B_{xx}(x,y) &= (B(x,y))_t\end{aligned} \tag{7}$$

Substituting (4) into the boundary condition of the equation (2), we obtain:

$$\begin{aligned}0 &= v_x(0,t) - b_2 v_t(0,t) \\ &= (a_2 n(0,0) - m(0,0) - a_2 - b) u_t(0,t)\end{aligned} \tag{9}$$

To deal with the boundary control problem of the anti-stable system, we employ the following transformation invented by A.Smyshlyaev [3] (for known q):

$$v(x,t) = u(x,t) + \frac{a+b}{1+ab}(-au(0,t) + \int_0^x u_t(y,t)dy) \tag{10}$$

Differentiating (3) with respect to *x*, setting *x*=1, and using the boundary condition of the equation (1), we get the following controller:

$$f(t) = \frac{ka(a+b)}{1+ab} u(0,t) - ku(1,t) - \frac{a+b}{1+ab} u_t(1,t) - \frac{k(a+b)}{1+ab} \int_0^1 u_t(y,t)dy) \tag{11}$$

where k is the controller gain .

We used the Caputo definition for fractional derivative of any function $f(t)$, for m to be the smallest integer that exceeds α, the Caputo fractional derivative $\alpha > 0$ is define as:

$$ {}_0^C D_t^\alpha f(t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{f^{m+1}(\tau)}{(t-\tau)^\alpha} d\tau \tag{12}$$

Based on the definition of (5), the Laplace transform of the Caputo fractional derivative is

$$L\{{}_0 D_t^\alpha f(t)\} = s^\alpha f(s) - \sum_{k=0}^{m-1} s^{\alpha-1-k} \left[\frac{d^k f(t)}{dt^k}\right]_{t=0^+} \tag{13}$$

where $m-1 < \alpha < m$

3 Analysis Method Using Fractional Calculus

The control law (11) has been widely used in the boundary control of the anti-stable diffusion equation; its effectiveness when applied to the boundary control of fractional diffusion equation is still unknown. In this section, we study the performance and properties of the fractional-order boundary controller, the fractional-order boundary controller can be obtained by the integer-order boundary control law (11), The fractional controller is obtained from the standard controller by replacing the first-order time derivative with a fractional derivative of order β,:

$$f(t)=\frac{ka(a+b)}{1+ab}u(0,t)-ku(1,t)-\frac{a+b}{1+ab}D_t^{\beta}u(1,t)-\frac{k(a+b)}{1+ab}\int_0^1 u_t(y,t)dy) \tag{14}$$

Laplace transform method is based on the classical partial differential equations to solve the general method of fundamental solutions, one of the basic idea of this method is: first, using Laplace transform the equations into ordinary differential equations, then obtain the general solution of ordinary differential equations and using the initial conditions and boundary conditions of the coefficients of the general solution, and finally obtained by Laplace inverse transformation in order to determine the expression of the fundamental solution equation. The literature [8] proposed the fractional Laplace transform-based partial differential equation algorithm is integer order partial differential equation based on the proposed, but for fractional partial differential equations, we must apply the MATLAB Math Toolbox, and the numerical solution of Laplace inverse transform can be resolved. Here are the time fractional partial differential equations for initial boundary value problem of the implementation process, the specific algorithm described as follows:

1. Take the Laplace transform of the equation (1) with respect to t, according to our application of the definition of Caputo fractional calculus, so an ODE is obtained for the transformed variable $u(x,s)$.

2. Solve the ODEs for $u(x,s)$ as a function of x, with the transform variable s still appearing as a parameter in the solution, and use the boundary conditions of the original problem to determine the precise form of $u(x,s)$.

3. Take the inverse Laplace transform of $u(x,s)$ with respect to s to the solution $u(x,t)$.

Several problems make the above method hard to use in practice to solve a PDE boundary control problem.

To test if fractional order boundary controllers can be used to stabilize the fractional wave equation, First, we choose k = 1 and study the responses the following two different systems were simulated. The case 1 is the integer-order vibration system using the fractional-order boundary controller. The case 2 is the fractional-order vibration system using the fractional-order boundary controller.

Case 1: $\alpha=1, \quad k=1, \quad \beta=0.5, 0.7, 0.9$,

Case 2: $\alpha=0.8, \quad k=1, \quad \beta=0.5, 0.7, 0.9$

The simulation results of the displacement response at the free end of integer-order string systems are shown in Figure 1, and the simulation results of the displacement response at the free end of fractional-order string systems are shown in Figure 2.

Simulation results show that the fractional order boundary controller to stabilize the string control system, the smaller fractional order controller will have a small overshoot and a relatively long rise time; and the larger fractional order controller has the overshoot and settling time is short features. So we can clude that t the fractional order controller to control anti-stable fractional order string system is better than the integer order controller.

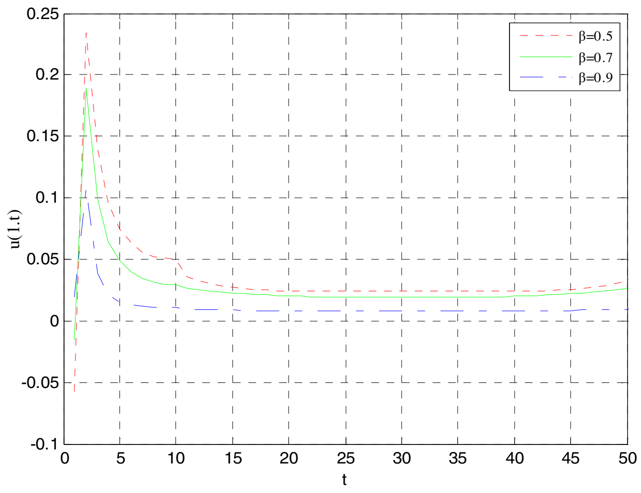


Fig. 1. Displacement of the tip end, $\alpha = 1$

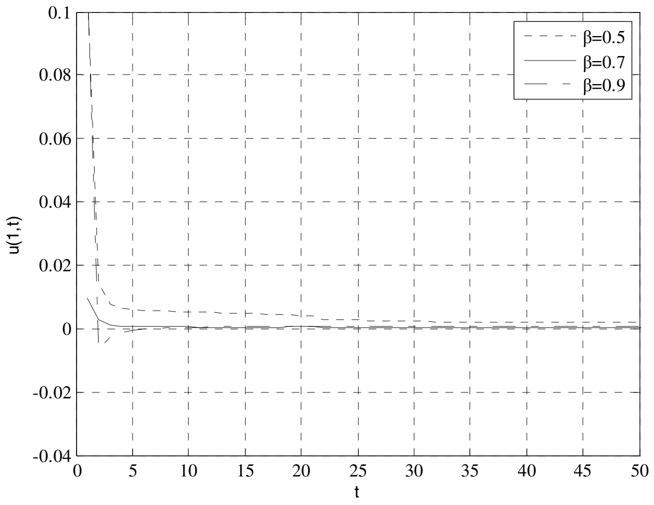


Fig. 2. Displacement of the tip end, $\alpha = 0.8$

4 Conclusions

The boundary controllers previously applied to the anti-able diffusion equation, simulation results show that the studied controllers are also applicable for the boundary control of diffusion equations. Boundary control of anti-stable diffusion equations is a new research topic. Based on the experience from the boundary control of the diffusion equations, controllers better suited for the fractional diffusion equations are to be explored in the future.

References

1. Chen, G.: Energy decay estimates and exact boundary value controllability for the diffusion equation in a bounded domain. J. Math. Pure Appl. 58, 249273 (1979)
2. Komornik, V., Zuazua, E.: A direct method for the boundary stabilization of the diffusion equation. J. Math. Pure Appl. 69, 33–54 (1990)
3. Krstic, M., Guo, B.-J., Balogh, A., Smyshlyaev, A.: Outputfeedback stabilization of an unstable diffusion equation. Automatica 44, 63–74 (2008)
4. Lasiecka, I., Triggiani, R.: Uniform stabilization of the diffusion equation with Dirichlet or Neumann feedback control without geometrical conditions. Appl. Math. Optim. 25, 189–224 (1992)
5. Smyshlyaev, A., Krstic, M.: Backstepping observers for a class of parabolic PDEs. Systems and Control Letters 54, 613–625 (2005)
6. Smyshlyaev, A., Krstic, M.: Closed form boundary state feedbacks for a class of 1-D partial integro-differential equations. IEEE Trans. on Automatic Control 49(12), 2185–2202 (2004)
7. Bardos, C., Halpern, L., Lebeau, G., Rauch, J., Zuazua, E.: Stabilization of the diffusion equation by Dirichlet type boundary feedback. Asymptotic Analysis 4(4), 285–291 (1991)
8. Beran, J.: Statistical methods for data with long-range dependence. Stastical Science 7, 404–427 (1992)
9. Darslaw, H.S., Jaeger, J.C.: Operational methods in Applied Mathematics. Dover, New York (1963)
10. Chen, Y., Moore, K.L.: Discretization schemes for fractional differentiators and integrators. IEEE Transactions on Circuits and System 1: Fundamental Theory and Applications 49(3), 363–367 (2002)
11. Feliu, V., Feliu, S.: A method of obtaining the time domain response of an equivalent circuit model. Journal of Electroanalytical Chemistry 435, 1–10 (1997)

Existence and Global Stability of Periodic Solutions of Generalized-Brain-State-in-a-Box (GBSB) Neural Models

Zhengxin Wang and Jinde Cao

Department of Mathematics, Southeast University,
Nanjing, 210096, China
jdcao@seu.edu.cn, zhengxinwang2008@gmail.com

Abstract. This paper aims to study the periodic solutions to two kinds of generalized brain-state-in-a-box (GBSB) neural models, which is different from the classical BSB neural networks. The results on the existence and global stability of the periodic solutions are derived.

Keywords: generalized brain-state-in-a-box, neural models, periodic solutions, stability.

1 Introduction

The brain-state-in-a-box (BSB) neural model was proposed by Anderson, Silverstein, Ritz, and Jones in 1977 as a memory model based on neurophysiological considerations [1], which dynamics is often described in discrete form as the following difference equation:

$$x(k+1) = g((I_n + \alpha W)x(k)),$$

where $x(k)$ is the state vector at time k, α is a step size, $W \in \mathbb{R}^{n\times n}$ is a weight matrix, and g is an activation function defined as following linear saturating function:

For every $y = (y_1, y_2, \cdots, y_n)^T \in \mathbb{R}^n$, then the i^{th} component of $g(y)$ is

$$g_i(y) = \begin{cases} 1, & \text{if } \ y_i \geq 1; \\ y_i, & \text{if } \ -1 < y_i < 1; \\ -1, & \text{if } \ y_i \leq -1. \end{cases}$$

Since the structure of the BSB neural model is similar to the nervous system, the BSB neural network can be used in the implementation of associative memories. Therefore the BSB neural model has been widely researched. Hui and Żak [2] studied the dynamics of the following generalized BSB (GBSB) neural models

$$x(k+1) = g((I_n + \alpha W)x(k) + \alpha b), \tag{1}$$

where $b \in \mathbb{R}^n$. In 1994, Lillo *et al.* [3] analyzed the dynamics of the GBSB neural network (1) and presented a synthesis procedure to realize an associative

© Springer-Verlag Berlin Heidelberg 2011

memory using the GBSB neural model. Later in 1995 Perfetti [4] presented qualitative properties of the BSB neural network and proposed an efficient synthesis procedure for networks that function as associative memories. Sometime after that, the BSB model was often studied by many researchers, see [5–8] in the reference. Recently, Park [9] presented a new optimization approach to the design of associative memories via the BSB neural network, which provided the large and uniform domains of attraction of the prototype patterns, the large robustness margin for the weight matrix of the perturbed BSB neural network, the asymptotic stability of the prototype patterns, and the global stability of the BSB neural network. On the whole, the study of BSB neural models has never stopped.

It is noted that the above results are all on the equilibrium point of the BSB neural networks. In this paper, we investigate the periodic solutions to the following GBSB neural model

$$x(k+1) = a(k)g((I_n + \alpha W)x(k)), \tag{2}$$

where $a : \mathbb{Z} \to \mathbb{R}^+$ is a ω periodic, presents the connection strength. The other notations are the same as above.

So far, many researchers have paid attention to the existence and stability of periodic solutions for neural networks or other mathematical models by Mawhin continuation theorem [10], see [11–15] and references therein. For example, Liu and Cao [12] studied the existence and stability of the unique periodic solution for the neural networks by using the differential inclusions theory, the Lyapunov-Krasovskii functional method and linear matrix inequality (LMI) technique. By applying the continuation theorem, Li [14] studied the existence and global stability of periodic solutions for a kind of the discrete cellular neural networks with delays.

In this paper, stimulated by the above studies, we study the existence and global stability of the GBSB neural model (2). The remainder is organized as follows. In section II, the result on the existence of periodic solutions of GBSB neural model (2) is obtained after introducing some notations and Mawhin continuation theorem. Finally, the stability of periodic solutions is analyzed in section III.

2 Existence of Periodic Solutions

In this section, we investigate the existence of periodic solutions of the GBSB neural networks (2) by Mawhin continuation theorem [10]. Firstly we introduce some notations and lemmas.

2.1 Preliminaries

For convenience of recounting, we introduce some definitions as follows.

Let X and Y be real Banach spaces and $L : D(L) \subset X \to Y$ be a Fredholm operator with index zero; here $D(L)$ denotes the domain of L. This means that

ImL is closed in Y and $\dim \ker L = \dim(Y/\mathrm{Im}L) < +\infty$. Consider the supplementary subspaces X_1, Y_1 of X, Y respectively, such that $X = \ker L \bigoplus X_1$ and $Y = \mathrm{Im}L \bigoplus Y_1$, and let $P : X \to \ker L$ and $Q : Y \to Y_1$ be natural projectors. Clearly, $\ker L \cap (D(L) \cap X_1) = \{0\}$, thus the restriction $L_P := L|_{D(L)\cap X_1}$ is invertible. Denote by K the inverse of L_P. Let Ω be an open bounded subset of X with $D(L) \cap \Omega \neq \emptyset$. A map $N : \overline{\Omega} \to Y$ is said to be L-compact in $\overline{\Omega}$, if $QN(\overline{\Omega})$ is bounded and the operator $K(I - Q)N : \overline{\Omega} \to X$ is compact.

Now we introduce the Mawhin's continuation theorem which the study on the existence of periodic solutions based on.

Lemma 1 (Gaines and Mawhin [10]). Suppose that X and Y are two Banach spaces, and $L : D(L) \subset X \to Y$ is a Fredholm operator with index zero. Furthermore, $\Omega \subset X$ is an open bounded set and $N : \overline{\Omega} \to Y$ is L-compact operator on $\overline{\Omega}$. If
(1)$Lx \neq \lambda Nx, \forall x \in \partial\Omega \cap D(L), \lambda \in (0, 1)$;
(2)$Nx \notin \mathrm{Im}L, \forall x \in \partial\Omega \cap \ker L$;
(3)The Brouwer degree

$$\deg(JQN, \Omega \cap \ker L, 0) \neq 0,$$

where $J : \mathrm{Im}Q \to \ker L$ is an isomorphism. Then the equation $Lx = Nx$ has a solution in $\overline{\Omega} \cap D(L)$.

Here, we adopt the following symbols.

$\mathbb{R}$, $\mathbb{R}^+$, $\mathbb{Z}$, $\mathbb{R}^n$ and $\mathbb{R}^{n\times n}$ denote the real numbers, positive real numbers, integer numbers, n-dimension Euclidean space and $n \times n$ real matrix space, respectively. Let $I_\omega = \{0, 1, \cdots, \omega - 1\}$. As introduced in the literatures [13, 14] and so on, we introduce again the following notations here.

$$l_n = \{x = \{x(k)\} : x(k) \in \mathbb{R}^n, k \in \mathbb{Z}\}.$$

Let $l^\omega \subset l_n$ be the subset of all ω periodic sequences equipped with the norm $\|\cdot\|$ as

$$\|x\| = \sum_{i=1}^{n} \max_{k\in I\omega} |x_i(k)|,$$

where $x = \{x(k)\} = \{(x_1(k), x_2(k), \cdots, x_n(k)), k \in \mathbb{Z}\} \in l^\omega$. Then it is easy to show that l^ω is a finite-dimensional Banach space. Let

$$l_0^\omega = \left\{ x = \{x(k)\} \in l^\omega : \sum_{k=0}^{\omega-1} x(k) = 0 \right\},$$

$$l_c^\omega = \{x = \{x(k)\} \in l^\omega : x(k) = h \in \mathbb{R}^n, k \in \mathbb{Z}\},$$

then it is easy to note that l_0^ω and l_c^ω are both closed linear subspace of l^ω, moreover,

$$l^\omega = l_0^\omega \oplus l_c^\omega, \quad \dim l_c^\omega = n.$$

In order to apply lemma 1, we take $X = Y = l^\omega$ and denote a linear operator by

$$L : X \to Y,\ (Lx)(k) = x(k+1) - x(k),\ x \in X,\ k \in \mathbb{Z},$$

a nonlinear operator

$$N : X \to Y,\ (Nx)(k) = a(k)g(x(k) + \alpha W x(k)) - x(k),\ x \in X,\ k \in \mathbb{Z}.$$

It is easy to see that L is a bounded linear operator, furthermore, one has

$$\ker L = l_c^\omega,\ \mathrm{Im}L = l_0^\omega,\ \dim\ker L = \mathrm{codim}\ \mathrm{Im}L = n,$$

therefore L is a Fredholm operator with index zero.

Let projectors $P : X \to \ker L$ and $Q : Y \to \mathrm{Im}Q$ be defined by

$$Px = \frac{1}{\omega}\sum_{s=0}^{\omega-1} x(s),\ x \in X, \qquad Qy = \frac{1}{\omega}\sum_{s=0}^{\omega-1} y(s),\ y \in Y.$$

It is easy to verify that P and Q are continuous projectors such that $\mathrm{Im}P = \ker L$ and $\mathrm{Im}L = \ker Q$. Set the operator K_p be the generalized inverse of $L|_{D(L)\cap\ker P}$, then

$$K_p : \mathrm{Im}L \to D(L)\cap\ker P,\quad (K_p y)(k) = \sum_{s=0}^{k-1} y(s) - \frac{1}{\omega}\sum_{s=0}^{\omega-1}(\omega - s)y(s).$$

Therefore QN and $K_p(I-Q)N$ are continuous. Noting that X is a finite-dimensional Banach space, which together with the Arzela-Ascoli theorem, it is easy to verify that $QN(\bar{\Omega})$ and $K_p(I-Q)N\Omega)$ are both relatively compact for any open bounded set $\Omega \subset X$. Therefore, N is L-compact on $\bar{\Omega}$, here Ω is any open bounded set of X.

Remark 1. It follows from the definition of g that $|g_i(x) - g_i(y)| \le |x_i - y_i|,\ i = 1, 2, \cdots, n$, for all $x = (x_1, x_2, \cdots, x_n),\ y = (y_1, y_2, \cdots, y_n) \in \mathbb{R}^n$.

2.2 The Existence of Periodic Solutions

Based on the Mawhin continuation theorem and preliminaries introduced above, we have the following result.

Theorem 2. The generalized brain-state-in-a-box neural networks (2) has at least one ω-periodic solutions.

Proof. Let $\Omega_1 = \{x :\ Lx = \lambda Nx,\ \lambda \in (0,1)\}$. If $x(k) \in \Omega_1$, then it follows from the definitions of L and N that

$$x(k+1) - x(k) = \lambda a(k)g(x(k) + \alpha W x(k)) - \lambda x(k),$$

that is

$$x_i(k+1) - x_i(k) = \lambda a(k)g_i(x(k) + \alpha W x(k)) - \lambda x_i(k),\quad i = 1, 2, \cdots, n. \quad (3)$$

For each $i \in \{1, 2, \cdots, n\}$, it follows from the (5) that

$$\begin{aligned}\max_{k\in I_\omega}|x_i(k)| &= \max_{k\in I_\omega}|x_i(k+1)|\\ &\le \max_{k\in I_\omega}[(1-\lambda)|x_i(k)| + \lambda a(k)|g_i(x(k)+\alpha W x(k))|]\\ &\le (1-\lambda)\max_{k\in I_\omega}|x_i(k)| + \lambda \max_{k\in I_\omega} a(k)|g_i(x(k)+\alpha W x(k))|,\end{aligned}$$

moreover, we have

$$\max_{k\in I_\omega}|x_i(k)| \le \max_{k\in I_\omega} a(k)|g_i(x(k)+\alpha W x(k))| \le \max_{k\in I_\omega} a(k) := \bar{a}.$$

Therefore
$$\|x\| \le n\bar{a}, \quad x \in \Omega_1.$$

Denoting $\underline{a} = \dfrac{1}{\omega}\displaystyle\sum_{k=0}^{\omega-1} a(k)$, therefore, $\underline{a} \le \bar{a}$.

Now we take $\Omega = \{x = \{x(k)\} = \{(x_1(k), x_2(k), \cdots, x_n(k)), k \in \mathbb{Z}\} \in X : \max_{k\in I_\omega}|x_i(k)| \le 1 + \bar{a},\ i = 1, 2, \cdots, n\}$. Therefore, Ω satisfies condition (1) of lemma 1. If $x \in \partial\Omega \cap \ker L$, then $x \equiv h \in \mathbb{R}^n$ is constant vector. Thus for $x \in \partial\Omega \cap \ker L$, it follows from the definitions of Q and N that

$$\begin{aligned}QNx &= \sum_{k=0}^{\omega-1}\left(\frac{a(k)g(x+\alpha W x)}{\omega} - \frac{x}{\omega}\right)\\ &= \underline{a}g(x+\alpha W x) - x\\ &= \begin{pmatrix}\underline{a}g_1(x+\alpha W x) - x_1\\ \underline{a}g_2(x+\alpha W x) - x_2\\ \vdots\\ \underline{a}g_n(x+\alpha W x) - x_n\end{pmatrix} \ne 0,\end{aligned}$$

the last inequality hold as following reason:
Since $x \in \partial\Omega \cap \ker L$, there exists a $i \in \{1, 2, \cdots, n\}$ such that $|x_i| = 1 + \bar{a}$, therefore, $|\underline{a}g_i(x+\alpha W x) - x_i| \ge |x_i| - |\underline{a}g_i(x+\alpha W x)| \ge 1 + \bar{a} - \underline{a} > 0$.

Therefore Ω satisfies condition (2) of lemma 1.

Define $H(\mu, x) = -\mu x + (1-\mu)JQNx$, for $\mu \in [0,1],\ x \in X$, where J is an identity operator. Then for all $\mu \in [0,1],\ x \in \partial\Omega \cap \ker L,\ H(\mu, x) \ne 0$ holds similarly. Then by the degree theory

$$\begin{aligned}\deg\{JQN(\cdot), \Omega \cap \ker L, 0\} &= \deg\{H(0,\cdot), \Omega \cap \ker L, 0\}\\ &= \deg\{H(1,\cdot), \Omega \cap \ker L, 0\}\\ &= \deg\{-I, \Omega \cap \ker L, 0\} \ne 0.\end{aligned}$$

Applying lemma 1, we reach the conclusion. □

Corresponding to GBSB (1), the GBSB neural model

$$x(k+1) = a(k)g((I_n + \alpha W)x(k) + \alpha b), \tag{4}$$

has the similarly result as following corollary.

Corollary 3. The generalized brain-state-in-a-box neural networks (4) has at least one ω-periodic solutions.

3 Stability of Periodic Solution

In this section, we discuss the stability of the periodic solution by constructing suitable Lyapunov functions.

Theorem 4. The ω-periodic solution of generalized brain-state-in-a-box neural networks (2) is globally stable if $\delta = 1 - \bar{a}\big(1 + \alpha \sum_{i=1}^{n} \bar{\omega}_i\big) > 0$, where $\bar{\omega}_i = \max\limits_{1\le j\le n} |\omega_{ij}|$.

Proof. It follows from theorem 2 that (2) exists an ω-periodic solution, which denoted by $x^*(k) = \{(x_1^*(k), x_2^*(k), \cdots, x_n^*(k))\}$ here. To prove it is globally stable, let $x(k) = \{(x_1(k), x_2(k), \cdots, x_n(k))\}$ be an arbitrary solution of (2), and $u_i(k) = x_i(k) - x_i^*(k)$. Then

$$\begin{aligned} u_i(k+1) &= x_i(k+1) - x_i^*(k+1) \\ &= a(k) g_i((I_n + \alpha W)x(k)) - a(k) g_i((I_n + \alpha W)x^*(k)) \\ &= a(k)[g_i((I_n + \alpha W)x(k)) - g_i((I_n + \alpha W)x^*(k))]. \end{aligned}$$

Now define the function V as

$$V(k) = \sum_{i=1}^{n} |u_i(k)|.$$

$$\begin{aligned} \nabla V &= V(k+1) - V(k) = \sum_{i=1}^{n} |u_i(k+1)| - \sum_{i=1}^{n} |u_i(k)| \\ &= \sum_{i=1}^{n} |x_i(k+1) - x_i^*(k+1)| - \sum_{i=1}^{n} |x_i(k) - x_i^*(k)| \\ &= \sum_{i=1}^{n} a(k) |g_i((I_n + \alpha W)x(k)) - g_i((I_n + \alpha W)x^*(k))| - \\ &\quad \sum_{i=1}^{n} |x_i(k) - x_i^*(k)| \\ &\le a(k) \sum_{i=1}^{n} \left| x_i(k) + \alpha \sum_{j=1}^{n} w_{ij} x_j(k) - x_i^*(k) - \alpha \sum_{j=1}^{n} w_{ij} x_j^*(k) \right| - \\ &\quad \sum_{i=1}^{n} |x_i(k) - x_i^*(k)| \\ &\le [a(k) - 1] \sum_{i=1}^{n} |x_i(k) - x_i^*(k)| + \alpha a(k) \sum_{i=1}^{n} \sum_{j=1}^{n} |w_{ij}| |x_j(k) - x_j^*(k)| \end{aligned}$$

$$\leq [\bar{a}-1]\sum_{i=1}^{n}|x_i(k)-x_i^*(k)| + \alpha\bar{a}\sum_{i=1}^{n}\bar{\omega}_i\sum_{j=1}^{n}|x_j(k)-x_j^*(k)|$$
$$\leq \left[\bar{a}\left(1+\alpha\sum_{i=1}^{n}\bar{\omega}_i\right)-1\right]\sum_{i=1}^{n}|x_i(k)-x_i^*(k)|$$
$$= -\delta\sum_{i=1}^{n}|x_i(k)-x_i^*(k)| \leq 0. \tag{5}$$

The first inequality according to remark 1. Summing the both sides of (5) from 0 to $m-1$

$$V(m)+\delta\sum_{k=0}^{m-1}\sum_{i=1}^{n}|x_i(k)-x_i^*(k)| \leq V(0),$$

which implies

$$\sum_{k=0}^{\infty}\sum_{i=1}^{n}|x_i(k)-x_i^*(k)| \leq \delta^{-1}V(0) < \infty,$$

that is $\lim\limits_{k\to\infty}|x_i(k)-x_i^*(k)| = 0,\ i = 1,2,\cdots,n$. This completes the proof of Theorem 3. □

Similarly, one has the following result for GBSB neural model (4)

Corollary 5. The ω-periodic solution of generalized brain-state-in-a-box neural networks (4) is globally stable if $\delta = 1-\bar{a}\big(1+\alpha\sum_{i=1}^{n}\bar{\omega}_i\big) > 0$, where $\bar{\omega}_i = \max\limits_{1\leq j\leq n}|\omega_{ij}|$.

Remark 2. If periodic solution is global stability, then it is unique. Therefore, theorem 4 implies the uniqueness of periodic solution.

Remark 3. There is no limitation on the $a(k)$ on the existence of periodic solutions, however, the periodic solution is global stability if $\bar{a} < \frac{1}{1+\alpha\sum_{i=1}^{n}\bar{\omega}_i}$.

Remark 4. In [5], Varga, Elek and Żak proposed new type of neural network models, which can be viewed as discrete linear systems operating on compact, convex domains. They select the following nonlinear activation function Ψ : $\mathbb{R}^n \to \mathbb{R}^n$:

$$\Psi(y) = \begin{cases} y, & \text{if } y \in U, \\ z, & \text{if } y \notin U, \end{cases}$$

where $U \subset \mathbb{R}^n$ is a compact convex domain such that the origin belongs to it, z is the point of intersection of the boundary of U by the line segment joining $\mathbf{0} \in \mathbb{R}^n$ and y. This paper discussed the GBSB neural models on compact and convex domains by trying a different method of [5], by adding a connection strength of the activation function, and obtained results on the existence and stability of periodic solutions of GBSB neural model.

4 Conclusion

This paper investigates the periodic solutions for two kinds of generalized brain-state-in-a-box (GBSB) neural models which are different from the previous literatures. The results on the existence and global stability of two kinds of GBSB neural models are obtained.

Acknowledgements. This work was jointly supported by the National Natural Science Foundation of China under Grant No. 60874088, the Specialized Research Fund for the Doctoral Program of Higher Education under Grant 20070286003, the Scientific Research Foundation of Graduate School of Southeast University under Grant YBJJ1028, the JSPS Innovation Program under Grant CX10B_060Z, and the NSF of Anhui Province No. 10040606Q01.

References

1. Anderson, J.A., Silverstein, J.W., Ritz, S.A., Jones, R.S.: Distinctive Features, Categorical Perception and Probability Learning: Some Applications of a Neural Model. In: Anderson, J.A., Rosenfeld, E. (eds.) Neurocomputing: Foundations of Research. MIT Press, Cambridge (1988)
2. Hui, S., Żak, S.H.: Dynamical Analysis of the Brain-State-in-a-Box (BSB) Neural Models. IEEE Trans. Neural Networks 3, 86–94 (1992)
3. Lillo, W.E., Miller, D.C., Hui, S., Żak, S.H.: Synthesis of Brain-State-in-a-Box (BSB) Based Associative Memories. IEEE Trans. Neural Networks 5, 730–737 (1994)
4. Perfetti, R.: A Synthesis Procedure for Brain-State-in-a-Box (BSB) Neural Networks. IEEE Trans. Neural Networks 6, 1071–1080 (1995)
5. Varga, I., Elek, G., Żak, S.H.: On the Brain-State-in-a-Convex-Domain Neural Models. Neural Networks 9, 1173–1184 (1996)
6. Park, J., Cho, H., Park, D.: On the Design of BSB Neural Associative Memories Using Semidefinite Programming. Neural Comput. 11, 1985–1994 (1999)
7. Park, J., Park, Y.: An Optimization Approach to Design of Generalized BSB Neural Associative Memories. Neural Comput. 12, 1449–1462 (2000)
8. Sevrani, F., Abe, K.: On the Synthesis of Brain-State-in-a-Box Neural Models with Application to Associative Memory. Neural Comput. 12, 451–472 (2000)
9. Park, Y.: Optimal and Robust Design of Brain-State-in-a-Box Neural Associative Memories. Neural Networks 23, 210–218 (2010)
10. Gaines, R.E., Mawhin, J.L.: Coincidence Degree and Nonliear Differential Equations. Springer, Beilin (1977)
11. Huang, H., Ho, D.W.C., Cao, J.D.: Analysis of Global Exponential Stability and Periodic Solutions of Neural Networks with Time-Varying Delays. Neural Networks 18, 161–170 (2005)
12. Liu, X.Y., Cao, J.D.: On Periodic Solutions of Neural Networks via Differential Inclusions. Neural Networks 22, 329–334 (2009)
13. Fan, M., Wang, K.: Periodic Solution of a Discrete Time Nonautonomous Ratio-Dependent Predator-Prey System. Math. Comput. Model 35, 951–961 (2002)
14. Li, Y.K.: Global Stability and Existence of Periodic Solutions of Discrete Delayed Cellular Neural Networks. Phys. Lett. A 333, 51–61 (2004)
15. Agarwal, R.P.: Difference Equations and Inequalities: Theory, Methods and Applications. Marcel Dekker, New York (2000)

Neural Network-Based Dynamic Surface Control of Nonlinear Systems with Unknown Virtual Control Coefficient*

Yingchun Wang, Guotao Hui, Zhiliang Wang, and Huaguang Zhang

Information Science and Engineering, Northeastern University,
Shenyang 110004, China
{drwangyc,zhg516}@gmail.com, hgt820112@163.com,
wangzhiliang@ise.neu.edu.cn

Abstract. This paper is concerned with the adaptive control problem for a class of strict-feedback nonlinear systems, in which unknown virtual control gain function is the main feature. Based on the neural network approximate ability and backstepping control design technique, adaptive neural network based dynamic surface control technique is developed. The advantage is that it does not require priori knowledge of virtual control gain function sign, which is usually demanded in many designs. At the same time, by dynamic surface control scheme, the explosion of computation is circumvented. The control performance of closed-loop systems is improved by adaptive modifying the estimated error upper bound. By theoretical analysis, the signals of closed-loop systems are globally ultimately bounded and the control error converges to a small residual set around the origin.

Keywords: Adaptive control, neural networks, dynamic surface control, strict-feedback systems, unknown virtual control gain function.

1 Introduction

Since the integral backstepping technology was proposed to deal with the nonlinear system control problem [1], it has been paid much attention [2–4]. In [2], systematic adaptive backstepping control techniques were introduced for nonlinear systems including strict-feedback systems, pure-feedback systems and block strict feedback systems. In [3], adaptive backstepping controller approach was extended to a class of nonlinear systems with stochastic jump parameters. For systems with high uncertainty, for example, the uncertainty that cannot be linearly parameterized or is completely unknown, various adaptive control methods were further developed in[5–13] by means of neural network or fuzzy logic system based backstepping techniques. In these papers, the backstepping technique

* This work was supported by the National Natural Science Foundation of China (60904017, 60804006), the Fundamental Research Funds for the Central Universities (N100404025).

© Springer-Verlag Berlin Heidelberg 2011

was used to synthesize adaptive control laws for strictly feedback systems or lower triangular systems. The controllers can achieve bounded tracking error for bounded initial states. The main advantages of backstepping technique are that many restriction of matching conditions for usual nonlinear control system are removed, such as matching condition, extended matching conditions, or growth conditions [14].

Though backstepping has become one of the most popular design methods for a large class of SISO nonlinear systems, it also subject to the problem of "explosion of complexity" in the traditional approaches, that is, the complexity of controller grows drastically as the order of the system increases. This "explosion of complexity" is caused by the repeated differentiations of certain nonlinear functions such as virtual controls, as pointed out in [15, 16]. To overcome such problem, dynamic surface control approach was proposed for a class of strict-feedback nonlinear systems with known system functions. By introducing first-order filtering of the synthetic virtual control input at each step of traditional backstepping approach, the system differentiation is not required and the smoothness of system functions need not be considered [11, 17, 18]. When the system functions are unknown, neural networks were used to approximate the unknown functions and adaptive dynamic surface control scheme was provided in [19–21].

On the other hand, virtual control gain function plays an important role in the controller design. Usually, it assumed that the priori knowledge about the signs of virtual control coefficients is known, which is either strictly positive or strictly negative. When this priori knowledge is unknown, that is, the signs of virtual control coefficients are unknown, it is a deeply challenge. The first solution was provided by Nussbaum for a class of first-order linear systems [22], where the Nussbaum-type gain function concept was originally proposed. This kind of control approach was extend to the high-frequency control gain linear systems [23, 24], in which the arguments of the Nussbaum functions are constructed. For the nonlinear systems, in [25], the high order nonlinear systems for constant virtual control coefficient was considered and the coupling terms in backstepping were canceled by young's inequality technique. In [26], the nonlinear time-delay system with unknown virtual control coefficient was investigated, and adaptive neural controller was designed. In [27], the pure-feedback nonlinear systems with unknown dead zone and perturbed uncertainties was studied based on the neural network model and dynamic surface control technique. In [28], such research was generalized to the stochastic system and fuzzy adaptive controller was designed such that closed-loop stochastic systems are globally ultimately bounded in probability. To the best of authors knowledge, the results on dynamic surface control for unknown virtual control coefficients systems have not been found in the literature, which motivates our study of this paper.

In this paper, we consider adaptive dynamic surface control problem for a class of nonlinear systems with the unknown virtual control coefficient. Based on the neural network approximate ability, a new neural network-based adaptive dynamic surface control design scheme is proposed. It avoids the priori knowledge assumption of the virtual control gain function sign, and the solving problem due to the "explosion of complexity". Furthermore, the control performance of closed-loop systems can be improved through the adaptive adjusting of estimated error upper limit. It is proved that all the signals in the closed-loop systems are globally ultimately bounded.

2 Problem Formulation

Consider the class nonlinear systems described below

$$\begin{aligned}\dot{x}_i(t) =& f_i(\bar{x}_i)x_{i+1} + g_i(\bar{x}_i), i = 1, ..., n-1,\\ \dot{x}_n(t) =& f_n(x)u + g_n(x),\end{aligned} \tag{1}$$

where $t \geq 0$, $x = [x_1, \cdots, x_n]^T \in R^n$ is the state vector, and $\bar{x}_i = [x_1, \cdots, x_i]^T$, $\bar{x}_n = x$. u is the control input. Functions $f_i : R^i \to R$, $g_i : R^i \to R$ are unknown smooth, where $i = 1, ...n$. The aim of the paper is to design a globally stabilization controller such that all the signals of closed-loop system are bounded.

Assumption 1. *Unknown functions $f_i(\bar{x}_i)$ is smooth, and there exist scalars $\underline{f_i}$ and known smooth functions $\bar{f}_i(\bar{x}_i)$ such that $0 < \underline{f_i} \leq |f_i(\bar{x}_i)| \leq \bar{f}_i(\bar{x}_i) < \infty$, $\forall \bar{x}_i \in R^i$.*

Definition 1. *[22] Any continuous function $Q(s) : R \longrightarrow R$ is a function of Nussbaum type, if it has the following properties:*

$$\lim_{s \longrightarrow +\infty} \sup \frac{1}{s} \int_0^s Q(\rho) d\rho = +\infty$$

$$\lim_{s \longrightarrow +\infty} \inf \frac{1}{s} \int_0^s Q(\rho) d\rho = -\infty$$

For example, the continuous functions $\rho^2 cos(\rho)$ and $e^{\rho^2} cos(\pi/2\rho)$ suffice.

Lemma 1. *[26] Let $V(\cdot)$ and $\rho(\cdot)$ be smooth functions defined on $[0, t_f\}$ with $V(t) > 0$, $\forall t \in [0, t_f\}$, and $N(\cdot)$ be a even smooth Nussbaum-type function. If the following inequality holds:*

$$V(t) \leq c_0 + e^{-c_1 t} \int_0^t f(x(\tau))Q(\rho)\dot{\rho} e^{c_1 \tau} d\tau + e^{-c_1 t} \int_0^t \dot{\rho} e^{c_1 \tau} d\tau, \quad \forall t \in [0, t_f\} \tag{2}$$

where constant $c_1 > 0$, $f(x(t))$ is a time-varying parameter which takes values in the unknown closed intervals $I := [l^-, l^+]$ with $0 \notin I$, and c_0 represents some suitable constant, then $V(t)$, $\rho(t)$ and $\int_0^t f(x(\tau))Q(\rho)\dot{\rho} e^{c_1 \tau} d\tau$ must be bound on $[0, t_f\}$.

In this paper, the following RBF neural network will be used to approximate any continuous function $\Phi(Z) : R^n \longrightarrow R$

$$\Phi(Z) = \theta^T \xi(Z) \tag{3}$$

where $\theta \in R^n$ for some integer N is called weight vector, and $\xi(Z) = [\varsigma_1, \cdots, \varsigma_n]$ is a vector valued function defined in R^n with ς_i chosen as commonly Gaussian function, i.e., $\varsigma_i(Z) = exp(-\|(Z-\tau_i)\|/2\sigma^2)$, $i = 1, \cdots, n$, where $\tau_i \in R^q$ is constant vectors called the center of the basis function, and σ is the width of the basis function. As indicated in [29], the neural network can approximate any continuous function $\Phi(Z) \in R^n$ to arbitrary accuracy.as

$$\Phi(Z) = \theta^{*T}\xi(Z) + \delta(Z), \forall Z \in R^n \tag{4}$$

where θ^* is the ideal constant weights, and $\delta(Z)$ is the network reconstruction error, satisfying $\|\delta(Z)\| \leq \eta^*$, η^* is known constant.

Since $\Phi(Z)$ is unknown, we need to estimate θ^* online. We use the notation $\hat{\theta}$ to estimation of θ^* with the estimation error $\tilde{\theta}$ and develop an adaptive law to update the parameter $\tilde{\theta}$.

3 Adaptive Controller Design

In this section, we will incorporate the DSC technique proposed in [16] into a neural network based adaptive control design scheme for the nth-order system described by (1). Similar to the traditional backstepping design method, the recursive design procedure contains n steps. let us design adaptive control laws based on the following coordinate transformation:

$$\begin{aligned} z_1 =& x_1, \\ z_i =& x_i - \alpha_{i-1}, i = 2, ..., n, \end{aligned} \tag{5}$$

where α_{i-1} is the output of first-order filter of virtual control input $\bar{\alpha}_{i-1}$, which is to define lately. Let $u = \alpha_n$ is the practical controller. z_i is the ith error surface of x_i and α_{i-1}.

Then we have

$$\begin{aligned} \dot{z}_1(t) =& f_1(x_1)x_2 + g_1(x_1), \\ \dot{z}_i(t) =& f_i(\bar{x}_i)x_{i+1} + g_i(\bar{x}_i) - \dot{\alpha}_{i-1}, i = 2, ..., n, \end{aligned} \tag{6}$$

Define the Lyapunov function as follows,

$$V_z = \frac{1}{2}\sum_{i=1}^{n} z_i^2 \tag{7}$$

Then we have

$$\begin{aligned} \dot{V}_z &= \sum_{i=1}^{n} z_i(f_i(\bar{x}_i)x_{i+1} + g_i(\bar{x}_i) - \dot{\alpha}_{i-1}) \\ &= \sum_{i=1}^{n} z_i(f_i(\bar{x}_i)(z_{i+1} + \alpha_i) + g_i(\bar{x}_i) - \dot{\alpha}_{i-1}) \\ &\leq \sum_{i=1}^{n} z_i(f_i(\bar{x}_i)\alpha_i + \Phi_i(Z_i)) \end{aligned} \tag{8}$$

where
$\Phi_1(Z_1) = g_1(x_1) + z_1^2 - \dot{\alpha}_0$,
$\Phi_i(Z_i) = g_i(\bar{x}_i) - \dot{\alpha}_{i-1} + z_i^2 + f_{i-1}^2(\bar{x}_{i-1})z_i^2$,
$\Phi_n(Z_n) = g_n(\bar{x}_n) - \dot{\alpha}_{n-1} + f_{n-1}^2(\bar{x}_{n-1})z_n^2$
with $Z_1 = [x_1, \alpha_0, \dot{\alpha}_0] \in \Omega_{Z_1} \in R^3$, where Ω_{Z_1} is compact set.
Similarly,
$Z_i = [\bar{x}_i^T, \alpha_{i-1}, \dot{\alpha}_{i-1}] \in \Omega_{Z_i} \in R^{i+2}$
$Z_n = [\bar{x}_n^T, \alpha_{n-1}, \dot{\alpha}_{n-1}] \in \Omega_{Z_n} \in R^{n+3}$.

In equation (8), the result of inequality $2z_i f_i(\bar{x}_i)z_{i+1} \leq z_i^2 + f_i^2(\bar{x}_i)z_{i+1}^2$ is used.

Due to the unknown functions f_i and g_i, the controllers including $\Phi_i(Z_i)$ does not be applied directly. So, we use the RBF neural network to approximate the function $\Phi_i(Z_i)$ as follows:

$$\Phi_i(Z_i) = \theta_i^T \xi_i(Z_i) = \theta_i^{*T} \xi_i(Z_i) + \delta_i(Z_i) \tag{9}$$

Choose a virtual control $\bar{\alpha}_1$ as follows:

$$\bar{\alpha}_1 = Q(\rho_1)[k_1 z_1 + \hat{\theta}_1^T \xi_1(Z(1))] \tag{10}$$

with parameter adaptive update law

$$\dot{\rho}_1 = k_1 z_1^2 + \hat{\theta}_1^T \xi_1(Z(1))z_1 \tag{11}$$

and network weight update law

$$\dot{\hat{\theta}}_1 = \Gamma_1 \xi_1(Z(1))z_1 - \varrho_1 \Gamma_1 \hat{\theta}_1 \tag{12}$$

where constants k_1, Γ_1, and ϱ_1 are positive.

Let $\bar{\alpha}_1$ pass through a first-order filter with time constant ϵ_1 to obtain α_1

$$\epsilon_1 \dot{\alpha}_1 + \alpha_1 = \bar{\alpha}_1, \alpha_1(0) = \bar{\alpha}_1(0) \tag{13}$$

Similar, choose the virtual control $\bar{\alpha}_{i-1}$ as follows:

$$\bar{\alpha}_{i-1} = Q(\rho_{i-1})[k_{i-1} z_{i-1} + \hat{\theta}_{i-1}^T \xi_{i-1}(Z(i-1))] \tag{14}$$

with

$$\dot{\rho}_{i-1} = k_{i-1} z_{i-1}^2 + \hat{\theta}_{i-1}^T \xi_{i-1}(Z(i-1))z_{i-1} \tag{15}$$

and

$$\dot{\hat{\theta}}_{i-1} = \Gamma_{i-1} \xi_{i-1}(Z(i-1))z_{i-1} - \varrho_i \Gamma_{i-1} \hat{\theta}_{i-1} \tag{16}$$

where constants $k_{i-1}, \Gamma_{i-1}, \varrho_{i-1}$ are positive.

Let $\bar{\alpha}_{i-1}$ pass through a first-order filter with time constant ϵ_{i-1} to obtain α_{i-1}

$$\epsilon_{i-1} \dot{\alpha}_{i-1} + \alpha_{i-1} = \bar{\alpha}_{i-1}, \alpha_{i-1}(0) = \bar{\alpha}_{i-1}(0) \tag{17}$$

Finally, let the final control u be as follows:

$$u = \alpha_n = Q(\rho_n)[k_n z_n + \hat{\theta}_n^T \xi_n(Z(n)) + \hat{\eta}_n \text{sign}(z_n)] \tag{18}$$

with

$$\dot{\rho}_n = k_n z_n^2 + \hat{\theta}_n^T \xi_n(Z(n)) z_n + \hat{\eta}_n \text{sign}(z_n) z_n \tag{19}$$

$$\dot{\hat{\theta}}_n = \varGamma_n \xi_n(Z(n)) z_n - \varrho_n \varGamma_n \hat{\theta}_n \tag{20}$$

and network approximation error update law $\hat{\eta}_n$, which is the estimation of reconstruction error η_n

$$\dot{\hat{\eta}}_n = \kappa \left[|z_n| - \gamma \hat{\eta}_n \right]. \tag{21}$$

4 Stability Analysis

In this section, the semi-global boundedness of all the signals in the closed-loop system with be proven. To prove the boundedness, define the boundary layer errors as follows:

$$\begin{aligned} S_1 =& \alpha_1 - \bar{\alpha}_1 \\ S_{i-1} =& \alpha_{i-1} - \bar{\alpha}_{i-1} \end{aligned} \tag{22}$$

Moreover,

$$\begin{aligned} z_1 =& x_1 - y_d \\ z_i =& x_i - S_{i-1} - \bar{\alpha}_{i-1}, i = 2, \cdots, n-1 \end{aligned} \tag{23}$$

Then the closed-loop system in the new coordinates $z_i, \alpha_i, \tilde{\theta}_i$ can be expressed as follows:

$$\begin{aligned} \dot{z}_1 =& f_1(x_1)(z_2 + \alpha_1) + g_1(x_1), \\ \dot{z}_i =& f_i(\bar{x}_i)(z_{i+1} + \alpha_i) + g_i(\bar{x}_i) - \dot{\alpha}_{i-1}, i = 2, ..., n-1, \\ \dot{z}_n =& f_n(\bar{x}_n) Q(\rho_n)[k_n z_n + \hat{\theta}_n^T \xi_n(Z(n)) + \hat{\eta}_n \text{sign}(z_n)] + g_n(\bar{x}_n) - \dot{\alpha}_{n-1} \end{aligned} \tag{24}$$

Noting that

$$\dot{\alpha}_i = \frac{\bar{\alpha}_i - \alpha_i}{\epsilon_i} = -\frac{S_i}{\epsilon_i} \tag{25}$$

and we have

$$\begin{aligned} \dot{S}_i &= \dot{\alpha}_i - \dot{Q}(\rho_i)\dot{\rho}_i[k_i z_i + \hat{\theta}_i^T \xi_i(Z(i))] - Q(\rho_i)[k_i \dot{z}_i + \dot{\hat{\theta}}_i^T \xi_i(Z(i)) + \hat{\theta}_i^T \dot{Z}_i^T \xi_i(Z_i)] \\ &= -\frac{S_i}{\epsilon_i} - \dot{Q}(\rho_i)\dot{\rho}_i[k_i z_i + \hat{\theta}_i^T \xi_i(Z(i))] - Q(\rho_i)[k_i \dot{z}_i + \dot{\hat{\theta}}_i^T \xi_i(Z(i)) + \hat{\theta}_i^T \dot{Z}_i^T \xi_i(Z_i)] \end{aligned} \tag{26}$$

and by induction some positive continuous function D_i, we have

$$|\dot{S}_i + \frac{S_i}{\epsilon_i}| \leq D_i(z_1, \cdots, z_i, \hat{\theta}_1, \cdots, \hat{\theta}_i, S_1, \cdots, S_i), i = 1, \cdots, n-1 \tag{27}$$

then $\dot{S}_i S_i \leq -\frac{S_i^2}{\varepsilon_i} + \bar{D}_i |S_i|$.

Now, we are in the position to give our main result in the follow theorem.

Theorem 1. *Consider the closed-loop systems (1), the controller (18), and the adaptation laws that are described as (11), (12), (15), (16), (19), (20), and (21) with constants $k_i = k_{i1} + k_{i2} > 0, \varGamma_i > 0, \varrho_i > 0$ for $i = 1, ..., n$, $\epsilon_l > 0$ for $l = 1, ..., n-1$ and $\kappa > 0, \gamma > 0$. Then, for the bounded initial conditions, all the signals of closed-loop systems are semi-globally uniformly ultimately bounded.*

Proof. Let us consider the following Lyapunov function candidate:

$$V = V_z + \frac{1}{2}\sum_{i=1}^{n-1} S_i^2 + \frac{1}{2}\sum_{i=1}^{n} \varGamma_i^{-1}\tilde{\theta}_i^2 + \frac{1}{2}\kappa^{-1}\tilde{\eta}_n^2 \tag{28}$$

Then the time derivative of V is

$$\begin{aligned}
\dot{V} =& \dot{V}_z + \sum_{i=1}^{n-1} S_i \dot{S}_i + \sum_{i=1}^{n} \tilde{\theta}_i \varGamma_i^{-1} \dot{\hat{\theta}}_i + \tilde{\eta}_n \kappa^{-1} \dot{\hat{\eta}}_n \\
=& \sum_{i=1}^{n} z_i (f_i(\bar{x}_i)\alpha_i + \varPhi_i(Z_i)) + \sum_{i=1}^{n-1} S_i \dot{S}_i + \sum_{i=1}^{n} \tilde{\theta}_i \varGamma_i^{-1} \dot{\hat{\theta}}_i + \tilde{\eta}_n \kappa^{-1} \dot{\hat{\eta}}_n \\
=& \sum_{i=1}^{n-1} z_i (f_i(\bar{x}_i)(S_i + Q(\rho_i)[k_i z_i + \hat{\theta}_i^T \xi_i(Z(i))] + \varPhi_i(Z_i)) \\
& + \sum_{i=1}^{n} \{\dot{\rho}_i - k_i z_i^2 - \hat{\theta}_i^T \xi_i(Z(i)) z_i\} \\
& + z_n \{f_n(x) Q(\rho_n)[k_n z_i + \hat{\theta}_n^T \xi_n(Z(n)) + \hat{\eta}_n sign(z_n)] + \varPhi_n(Z_n)\} \\
& + \sum_{i=1}^{n-1} S_i \dot{S}_i + \sum_{i=1}^{n} \tilde{\theta}_i \left[z_i \xi_i(Z(i)) - \varrho_i \hat{\theta}_i \right] + \tilde{\eta}_n [|z_n| - \gamma \hat{\eta}_n] \\
\leq& \sum_{i=1}^{n-1} \left[f_i(\bar{x}_i) Q(\rho_i) \dot{\rho}_i + \dot{\rho}_i - k_i z_i^2 + \delta_i z_i \right] + \sum_{i=1}^{n-1} z_i f_i(\bar{x}_i) S_i \\
& - \sum_{i=1}^{n-1} \frac{S_i^2}{\epsilon_i} + \sum_{i=1}^{n-1} \bar{D}_i |S_i| - \varrho_i \tilde{\theta}_i \hat{\theta}_i - \gamma \tilde{\eta}_n \hat{\eta}_n \\
& + f_n(x) Q(\rho_n) \dot{\rho}_n + \dot{\rho}_n - (k_n z_n^2 + \hat{\eta}_n \mathrm{sign}(z_n) z_n) + \eta_n z_n + \tilde{\eta}_n |z_n| .
\end{aligned} \tag{29}$$

Consider the following inequalities results:

$$z_i f_i(\bar{x}_i) S_i \leq \frac{1}{2} z_i^2 + \frac{1}{2} f_i^2(\bar{x}_i) S_i^2 \leq \frac{1}{2} z_i^2 + \frac{1}{2} \bar{f}_i^2(\bar{x}_i) S_i^2, i = 1, ..., n-1, \tag{30}$$

$$\bar{D}_i |S_i| \leq \frac{1}{2} \bar{D}_i^2 + \frac{1}{2} S_i^2, \quad i = 1, ..., n-1, \tag{31}$$

$$-\varrho_i\tilde{\theta}_i^T\hat{\theta}_i \leq -\frac{1}{2}\varrho_i\left|\tilde{\theta}_i\right|^2 + \frac{1}{2}\varrho_i\left|\theta_i^*\right|^2, \quad i=1,...,n, \tag{32}$$

$$-\gamma\tilde{\eta}_n\hat{\eta}_n \leq -\frac{1}{2}\gamma\left|\tilde{\eta}_n\right|^2 + \frac{1}{2}\gamma\left|\eta_n^*\right|^2, \tag{33}$$

and let $k_i = k_{i1} + k_{i2}$, where $k_{i1} > 0$ and $k_{i2} > 0$, we have

$$-k_{i1}z_i^2 + \delta_i z_i \leq \frac{1}{4k_{i1}}\delta_i^2 \leq \frac{1}{4k_{i1}}\eta_i^{*2}, \quad i=1,...,n-1. \tag{34}$$

Then we obtain

$$\begin{aligned}\dot{V}(t) \leq& \sum_{i=1}^{n-1}\left[f_i(\bar{x}_i)Q(\rho_i)\dot{\rho}_i + \dot{\rho}_i - k_{i2}z_i^2 + \frac{1}{4k_{i1}}\eta_i^{*2}\right] \\ &+ \sum_{i=1}^{n-1}\left[\frac{1}{2}z_i^2 + \frac{1}{2}\bar{f}_i^2(\bar{x}_i)S_i^2\right] - \sum_{i=1}^{n-1}\frac{S_i^2}{\epsilon_i} + \sum_{i=1}^{n-1}\left[\frac{1}{2}\bar{D}_i^2 + \frac{1}{2}S_i^2\right] \\ &+ \sum_{i=1}^{n}\left[-\frac{1}{2}\varrho_i|\tilde{\theta}_i|^2 + \frac{1}{2}\varrho_i\left|\theta_i^*\right|^2\right] - \frac{1}{2}\gamma\left|\tilde{\eta}_n\right|^2 + \frac{1}{2}\gamma\left|\eta_n^*\right|^2 \\ &+ f_n(x)Q(\rho_n)\dot{\rho}_n + \dot{\rho}_n - k_n z_n^2 \\ =& \sum_{i=1}^{n}\{-cV_{zi} + f_i(\bar{x}_i)Q(\rho_i)\dot{\rho}_i + \dot{\rho}_i + d_i\}\end{aligned} \tag{35}$$

where $c = \min\{-2k_{i2}+1, 2k_n, \varrho_i/\Gamma_i, \gamma/\kappa, 2\epsilon_i^{-1} - \bar{f}_i^2 - 1\}$ with $2\epsilon_i^{-1} - \bar{f}_i^2 - 1 > 0$ and $d_i = \frac{1}{4k_{i1}}\eta_i^{*2} + \frac{1}{2}\varrho_i\left|\theta_i^*\right|^2 + \frac{1}{2}\bar{D}_i^2$, $d_n = \frac{1}{2}\gamma\left|\eta_n^*\right|^2 + \frac{1}{2}\varrho_n\left|\theta_n^*\right|^2$.

Multiplying (35) by e^{ct}, we have

$$d(e^{ct}V) \leq \sum_{i=1}^{n}\{f_i(\bar{x}_i)Q(\rho_i)\dot{\rho}_i + \dot{\rho}_i + d_i\}dt \tag{36}$$

Integrating (36) over $[0,T]$, we have

$$\begin{aligned}V(T) \leq& e^{-cT}V(0) + \sum_{i=1}^{n}d_i/c \\ &+ e^{-cT}\sum_{i=1}^{n}\int_0^T f_i(\bar{x}_i(\tau))Q(\rho_i)\dot{\rho}_i e^{c\tau}d\tau + e^{-cT}\sum_{i=1}^{n}\int_0^T \dot{\rho}_i e^{c\tau}d\tau\end{aligned} \tag{37}$$

From Lemma 1, we can draw the conclusion that for every $x_0 \in R^n$ function ρ_i, $e^{-cT}\int_0^T f_i(\bar{x}_i(\tau))Q(\rho_i)\dot{\rho}_i e^{c\tau}d\tau$ and $V(T)$ must be bounded. Moreover, all the signals of closed-loop systems are bounded.

5 Conclusion

In this paper, adaptive dynamic surface control technique was developed for a class of strict-feedback nonlinear systems in which the virtual control gain function is unknown. Backstepping based control technique was developed such that priori knowledge of virtual control gain function sign does not require. The problem of "explosion of complexity" was dealt. At the same time, all the signals of closed-loop systems are bounded and the control performance of systems is improved by the approach of adaptive modifying the estimated error upper bound.

References

1. Kanellakipoulos, I., Kokotovic, P.V., Morse, A.S.: Systematic Design of Adaptive Controllers for Feedback Linearizable Systems. IEEE Trans. Automat. Contr. 36(11), 1241–1253 (1991)
2. Krstic, M., Kanellakopoulos, I., Kokotovic, P.V.: Nonlinear and Adaptive Control Design. Wiley, New York (1995)
3. Xia, Y., Fu, M., Shi, P., Wu, Z., Zhang, J.: Adaptive Backstepping Controller Design for Stochastic Jump Systems. IEEE Trans. Automat. Contr. 54(12), 2853–2859 (2009)
4. Polycarpou, M.M., Mears, M.J.: Stable Adaptive Tracking of Uncertain Systems Using Nonlinear Parameterized Online Approximators. Int. J. Control 70(3), 363–384 (1998)
5. Ge, S.S., Wang, J.: Robust Adaptive Neural Nontrol for a Class of Perturbed Strict Feedback Nonlinear Systems. IEEE Trans. Neural Networks 13, 1409–1419 (2002)
6. Chen, W., Jiao, L., Li, J., Li, R.: Adaptive NN Backstepping Output-Feedback Control for Stochastic Nonlinear Strict-Feedback Systems With Time-Varying Delays. IEEE Trans. Syst., Man, and CyberN., Part B: Cybern. 40(3), 939–950 (2010)
7. Li, Y., Qiang, S., Zhuang, X., Kaynak, O.: Robust and Adaptive Backstepping Control for Nonlinear Systems Using RBF Neural Networks. IEEE Trans. Neural Networks 15(3), 693–701 (2004)
8. Chen, W., Zhang, Z.: Globally Stable Adaptive Backstepping Fuzzy Control for Output-feedback Systems with Unknown High-frequency Gain Sign. Fuzzy Sets and Systems 161, 821–836 (2010)
9. Tong, S.C., He, X.L., Zhang, H.G.: A Combined Backstepping and Small-gain Approach to Robust Adaptive Fuzzy Output Feedback Control. IEEE Trans. Fuzzy Systems 17(5), 1059–1069 (2009)
10. Zhang, T., Ge, S.S., Hang, C.C.: Adaptive Neural Network Control for Strict-feedback Nonlinear Systems Using Backstepping Design. Automatica 36(12), 1835–1846 (2000)
11. Choi, J.Y., Stoev, J., Farrell, J.: Adaptive observer backstepping control using neural networks. IEEE Trans. Neural Networks 12(5), 1103–1112 (2001)
12. Wang, M., Chen, B., Liu, X.P., Shi, P.: Adaptive Fuzzy Tracking Control for a Class of Perturbed Strict-feedback Nonlinear Time-delay Systems. Fuzzy Sets and Systems 159(8), 949–967 (2008)
13. Zhang, T.P., Wen, H., Zhu, Q.: Adaptive Fuzzy Control of Nonlinear Systems in Pure Feedback Form Based on Input-to-State Stability. IEEE Trans. Fuzzy Systems 18(1), 80–93 (2010)
14. Narendra, K.S., Annaswamy, A.M.: Stable Adaptive Systems. Prentice-Hall, Englewood Cliffs (1989)
15. Yip, P.P., HedrickK, J.K.: Adaptive Dynamic Surface Control: a Simplied Algorithm for Adaptive Backstepping Control of Nonlinear Systems. Int. J. Control 71(5), 959–979 (1998)

16. Swaroop, D., Hedrick, J.K., Yip, P.P., Gerdes, J.C.: Dynamic Surface Control for a Class of Nonlinear Systems. IEEE Trans. Automat. Contr. 45(10), 1893–1899 (2000)
17. Song, B., Hedrick, J.K.: Observer-based Dynamic Surface Control for a Class of Nonlinear Systems an LMI Approach. IEEE Trans. Automat. Contr. 49(11), 1995–2001 (2004)
18. Yoo, S.J., Park, J.B., Ho Choi, Y.: Adaptive Dynamic Surface Control for Stabilization of Parametric Strict-Feedback Nonlinear Systems With Unknown Time Delays. IEEE Trans. Automat. Contr. 52(12), 2360–2365 (2007)
19. Yoo, S.J., Park, J.B., Ho Choi, Y.: Adaptive Output Feedback Control of Flexible-Joint Robots Using Neural Networks: Dynamic Surface Design Approach. IEEE Trans. Neural Networks 19(10), 1712–1726 (2008)
20. Zhang, T.P., Ge, S.S.: Direct Adaptive NN Control of Nonlinear Systems in Strict-Feedback Form Using Dynamic Surface Control. In: 22nd IEEE International Symposium on Intelligent Control Part of IEEE Multi-Conference on Systems and Control, Singapore, pp. 315–320 (2007)
21. Wang, D., Huang, J.: Neural Network-Based Adaptive Dynamic Surface Control for a Class of Uncertain Nonlinear Systems in Strict-Feedback Form. IEEE Trans. Neural Networks 16(1), 195–202 (2005)
22. Nussbaum, R.D.: Some Remarks on a Conjecture in Parameter Adaptive Control. Syst. Contr. Lett. 3(5), 243–246 (1983)
23. Ryan, E.P.: A Universal Adaptive Stabilizer for a Class of Nonlinear Systems. Syst. Contr. Lett. 16(3), 209–218 (1991)
24. Martensson, B.: Remarks on Adaptive Stabilization of First-order Non-linear Systems. Syst. Contr. Lett. 14, 1–7 (1990)
25. Ye, X., Jiang, J.: Adaptive Nonlinear Design Without a Priori Knowledge of Control Directions. IEEE Trans. Automat. Contr. 43, 1617–1621 (1998)
26. Ge, S.S., Hong, F., Lee, T.H.: Adaptive Neural Control of Nonlinear Time-Delay Systems with Unknown Virtual Control Coefficients. IEEE Trans. Syst., Man, and CyberN., Part B: Cybern. 34(1), 499–516 (2004)
27. Zhang, T.P., Ge, S.S.: Adaptive Dynamic Surface Control of Nonlinear Systems with Unknown Dead Zone in Pure Feedback Form. Automatica 44, 1895–1903 (2008)
28. Wang, Y.C., Zhang, H.G., Wang, Y.Z.: Fuzzy Adaptive Control of Stochastic Nonlinear Systems with Unknown Virtual Control Gain Function. ACTA Automatica Sinica 32(2), 170–179 (2006)
29. Park, J., Sandberg, I.W.: Universal Approximation Using Radial-basis Function Networks. Neural Computation 3, 246–257 (1991)

Stochastic Stability Analysis of Delayed Hopfield Neural Networks with Impulse Effects

Wenfeng Hu, Chuandong Li, Sichao Wu, and Xiaofeng Liao

College of Computer, Chongqing University,
Chongqing 400030, China
licd@cqu.edu.cn

Abstract. This paper studies the global exponential stability of the delayed Hopfield neural networks with both stochastic perturbations and impulse effects. By means of the $It\hat{o}$ formula, Lyapunov-Razumikhin theorems and certain inequality techniques, some sufficient criteria are obtained which guarantee the global exponential stability of the delayed Hopfield neural networks with stochastic perturbations and impulse effects. The results characterize the intricate effects of the impulses and then can be used to estimate the feasible upper bounds of impulses. Furthermore, a numerical simulation is given to illustrate the validity of our results.

Keywords: global exponential stability, delayed Hopfield neural networks (NN), stochastic, impulse effects, Lyapunov-Razumikhin theorem, $It\hat{o}$ formula.

1 Introduction

In the past decades, Hopfield neural networks (HNN), proposed by Hopfield in the 1980s [1-2], has attracted considerable attention due to its widely employed in many areas such as pattern recognition, associate memories and combinatorial optimization (see [3-5] and references therein). To the best of authors' knowledge, in neural processing and signal transmission, there are several causes as the sources of instability, which may lead to bad performance in the applications mentioned above.

Among the sources of instability, time delays are inevitable in most circuits during the processing and transmission of signals. Hopfield neural networks with such time delays is called Delayed Hopfield neural networks (DHNN), which can be represented by delayed differential equations. Besides time delays effects, in the implementation of electronic neural networks, the state of neural networks is often subject to instantaneous perturbations and experience abrupt changes at certain moments of time which may be caused by a switching phenomenon, frequency change or other sudden noise; namely, the networks exhibit impulse effects. They frequently occur in fields such as economics, mechanics, electronics, telecommunications, medicine and biology, etc. It has also been known that impulse effects may lead to complicated results [6-9], for instance, it may cause instability of the previous system. On the other hand, impulsive control has been used for stabilization of the impulse-free neural networks which are even not asymptotically stable. Thus, the consideration of impulse effects is essential in our study.

© Springer-Verlag Berlin Heidelberg 2011

It is worth noting that, 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. Such factors are stochastic effects, which could stabilize or destabilize some certain neural networks. Hence, the stability analysis of stochastic neural networks becomes increasingly significant, and some results related to this subject have been published recently, see references [10-12]. However, most of them utilized the Lyapunov functional method. In this paper, we use the Lyapunov-Razumikhin method ([13-14]), by which it is much easier to find an appropriate Lyapunov function than functional, thus it has the advantage in verifying conditions for most systems. Our results are then used to estimate the feasible upper bounds of impulses, which guarantee the global exponential stability of the delayed Hopfield neural networks with stochastic perturbations and impulse effects.

2 Problem Statement and Preliminaries

In this paper, we consider the impulsive DHNN, which is often described by the delayed differential equations as follows:

$$\dot{u}(t) = Cu(t) + A\kappa(u(t)) + B\eta(u(t-\tau)) + I \quad t \neq t_k \tag{1}$$

$$\Delta u(t_k) = I_k\left(u\left(t_k^-\right)\right) \quad k = 1, 2, \cdots, \tag{2}$$

Where the fist equation (1) describes the evolution processes of the neural networks in the form of continuous-time, $u(t) \in R^n$ denotes the state vector of neurons, $n \geq 2$ is the number of neurons, the diagonal matrix $C = diag(c_1, c_2, \cdots, c_n)$ is the state feedback coefficient with scalar $0 < |c_i| < 1$, $A = (a_{ij})_{n\times n}$ and $B = (b_{ij})_{n\times n}$ are two real matrices, which represent the network's connection weight matrix and delayed connection weight matrix, respectively, $I \in R^n$ represents constant external inputs, $\tau > 0$ stands for the time delay. $\kappa(u) = [\kappa_1(u_1), \kappa_2(u_2), \cdots, \kappa_n(u_n)]^T$ and $\eta(u) = [\eta_1(u_1), \eta_2(u_2), \cdots, \eta_n(u_n)]^T$ are the activation functions used in the ith neuron, which are usually assumed to satisfy the following inequalities (according to Morita ([15]) and Yoshizawa ([16]).

Assumption 1. $\kappa(\cdot)$ and $\eta(\cdot)$ are assumed to be Lipschitz-continuous, i.e., $|\kappa_i(x) - \kappa_i(y)| \leq l_i^f |x-y|$ and $|\eta_i(x) - \eta_i(y)| \leq l_i^g |x-y|$, thus, $L_f = diag(l_1^f, l_2^f, \cdots, l_n^f)$ and $L_g = diag(l_1^g, l_2^g, \cdots, l_n^g)$ are called Lipschitz coefficients.

$I_k : R^n \to R^n$ $(k = 1, 2, \cdots)$ indicate the state jump operators which satisfy the following assumption.

Assumption 2. there always exist positive numbers j_k such that $\left\|I_k(x)-I_k(y)\right\| \le j_k \left\|x-y\right\|$ for any $x, y \in R^n$

Without loss of generality, we let $u(t)=u(t,t_0,\varphi)$ be any solution of DHNN (1) with impulse (2), which means $u(t)$ satisfies the initial condition: $u_i(s)=\varphi_i(s)$, $s\in[t_0-\tau,t_0]$ where $\varphi_i:[t_0-\tau,t_0]\to R, i=1,2,\cdots,n$.

For simplicity, we always shift the equilibrium point $u^*=\left[u_1^*,\cdots,u_n^*\right]^T$ of the impulsive DHNN to the origin by making a transformation $x(t)=u(t)-u^*$. What's more, when taking the stochastic perturbations into consideration, we use the $\omega(t)=\left[\omega_1(t),...,\omega_n(t)\right]^T$ to denote an n-dimensional Brownian motion defined on a complete probability space $\left(\Omega,F,\{F_t\}_{t\ge0},P\right)$ with a filtration $\{F_t\}_{t\ge0}$ satisfying the usual conditions (i.e., it is right continuous and F_0 contains all P-null sets). Then, the previous system can be rewritten as:

$$\begin{cases} dx(t)=\left[Cx(t)+Af(x(t))+Bg(x(t-\tau))\right]dt+\sigma(t,x(t),x(t-\tau))d\omega(t) & t\in[t_{k-1},t_k) \\ x(t)=x(t_k^-)+J_k(x(t_k^-)) & t=t_k \\ x(s)=\phi(s) & s\in[t_0-\tau,t_0] \end{cases} \tag{3}$$

Where $f(x(t))=\kappa(x(t)+u^*)-\kappa(u^*)$ and $g(x(t-\tau))=\eta(x(t-\tau)+u^*)-\eta(u^*)$, which implies that $f(x(t))\le L_f x(t)$ and $g(x(t))\le L_g x(t)$, according to the assum - ption 1. $J_k(x)=I_k(x+u^*)-I_k(u^*)$ with the property $\|J_k(x)\|\le j_k\|x\|, k=1,2,\cdots$, implied by assumption 2. Besides, $\phi(s)=\varphi(s)-u^*$ in system (3). Thus, we will investigate the (3) instead of the previous DHNN.

Where $\sigma(\cdot)$ are assumed to satisfy the following inequality:

Assumption 3. there exist matrices $D_1>0$ and $D_2>0$ such that:

$$\begin{aligned} trace\left[\sigma^T(t,x(t),x(t-\tau))\cdot\sigma(t,x(t),x(t-\tau))\right] \\ \le x^T(t)D_1x(t)+x^T(t-\tau)D_2x(t-\tau) \end{aligned}$$

3 Main Results

Theorem 1. The zero solution of system (3) is said to be globally exponentially stable in the mean square provided that the following conditions are satisfied:

(1). There exist positive scalars ε_1, ε_2, $q>1$ and matrices $Q_1=Q_1^T>0$, $Q_2=Q_2^T>0$, $P=P^T>0$, such that:

$\Omega_1 = PC + C^T P + \varepsilon_1 PAQ_1A^T P + \varepsilon_1^{-1} L_f^T Q_1^{-1} L_f + \varepsilon_2 PBQ_2B^T P + D_1 + \alpha qP \le 0$ and $\Omega_2 = \varepsilon_2^{-1} L_g^T Q_2^{-1} L_g + D_2 - \alpha P \le 0$ whenever $V(t+s) \le qV(t)$ for $s \in [-\tau, 0]$, where $q \ge e^{\lambda(l+1)d}$, λ, l, d are defined later.

(2). There exist natural number $l \ge 1$, $d = \max_k (t_k - t_{k-1})$ and $\tau \le d$ satisfying $t_k - t_{k-1} \ge \tau / l$ for any $k = 1, 2, \cdots$

(3). Let the λ^* to be the exponential decay rate, $\lambda = 2\lambda^*$, $\beta = \lambda_M (P + PJ + J^T P + J^T PJ) \cdot \lambda_m^{-1}(P)$ which reflects the effects of impulse, with the following inequalities hold: $\ln \beta + 2\lambda d < 0$.

Proof. Consider the Lyapunov function $V(t) = x^T(t) Px(t)$

We consider the derivation of $V(t)$ along the trajectories of system (3), according to the *Itô* formula, i.e., $dV(t) = \frac{\partial V}{\partial t} dt + \frac{\partial V}{\partial x} dx + \frac{1}{2} \frac{\partial^2 V}{\partial x^2} (dx)^2$

We can calculate as follows:

$$dE\{V(t)\} = E\{LV(t)\} dt \tag{4}$$

Where $E\{V(t)\}$ means the expectation of the $V(t)$ and

$$\begin{aligned} LV(t) = {} & \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} \left[Cx(t) + Af(x(t)) + Bg(x(t-\tau)) \right] \\ & + \frac{1}{2} trace \left[\sigma^T (t, x(t), x(t-\tau)) \cdot \frac{\partial^2 V}{\partial x^2} \cdot \sigma(t, x(t), x(t-\tau)) \right] \end{aligned} \tag{5}$$

It's not difficult to see that:

$$2x^T(t) PAf(x(t)) \le \varepsilon_1 x^T(t) PAQ_1A^T Px(t) + \varepsilon_1^{-1} x^T(t) L_f^T Q_1^{-1} L_f x(t) \tag{6}$$

$$2x^T(t) PBg(x(t-\tau)) \le \varepsilon_2 x^T(t) PBQ_2B^T Px(t) + \varepsilon_2^{-1} x^T(t-\tau) L_g^T Q_2^{-1} L_g x(t-\tau) \tag{7}$$

When, for a constant $q \ge 1$, $V(t+s) \le qV(t)$ for all $s \in [-\tau, 0]$. So, for a constant $\alpha \ge 0$, we have

$$\alpha \left[qx^T(t) Px(t) - x^T(t-\tau) Px(t-\tau) \right] \ge 0 \tag{8}$$

Substituting (6), (7) and (8) into (5) yields

$$\begin{aligned} LV(t) \le {} & x^T(t) \left[PC + C^T P + \varepsilon_1 PAQ_1A^T P + \varepsilon_1^{-1} L_f^T Q_1^{-1} L_f + \varepsilon_2 PBQ_2B^T P + D_1 \right] x(t) \\ & + x^T(t-\tau) \left[\varepsilon_2^{-1} L_g^T Q_2^{-1} L_g + D_2 \right] x(t-\tau) \\ & + \alpha \left[qx^T(t) Px(t) - x^T(t-\tau) Px(t-\tau) \right] \le \eta^T(t,\tau) \begin{bmatrix} \Omega_1 & 0 \\ 0 & \Omega_2 \end{bmatrix} \eta(t,\tau) \end{aligned} \tag{9}$$

Where $\eta(t,\tau)=\left[x(t),x(t-\tau)\right]^T$, Ω_1 and Ω_2 are defined in the condition (1). Besides, we can know that, $\Omega_1 \le 0$ and $\Omega_2 \le 0$ lead to $\dfrac{dE\{V(t)\}}{dt}=E\{LV(t)\}\le 0$.

It's obvious to see that: $\lambda_1\|x\|^2 \le V(t) \le \lambda_2\|x\|^2$, where $\lambda_1=\lambda_m(P)$ and $\lambda_2=\lambda_M(P)$, we define that $\|\bar{\phi}\|_{(\tau)}=\sup_{-\tau\le s\le 0}\|\phi(s)\|$, we can choose $M\ge 1$ such that:

$$\lambda_2\|\bar{\phi}\|_{(\tau)}^2 < M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)} \le q\lambda_2\|\bar{\phi}\|_{(\tau)}^2 \tag{10}$$

Our goal is to prove that the next inequality is hold for any $k=1,2,\cdots$:

$$V(t)\le M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_k-t_0)} \qquad t\in[t_{k-1},t_k) \tag{11}$$

In order to reach the goal above, the mathematical induction method is applied later, when $k=1$, we firstly show that:

$$V(t)\le M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)} \qquad t\in[t_0,t_1) \tag{12}$$

From (10), for any $t\in[t_0-\tau,t_0]$, we get

$$V(t)\le \lambda_2\|x(t)\|^2 \le \lambda_2\|\bar{\phi}\|_{(\tau)}^2 < M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)} \tag{13}$$

Therefore, if (12) is not true, there always exist $\tilde{t}\in[t_0,t_1)$ such that

$$V(\tilde{t}) > M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)} > \lambda_2\|\bar{\phi}\|_{(\tau)}^2 \ge V(t_0+s),\ \ s\in[-\tau,0) \tag{14}$$

Then, we can assume $t^*=\min\left\{t\in(t_0,\tilde{t})\middle| V(t)\ge M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)}\right\}$

Obviously, $V(t^*)\ge M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)} > \lambda_2\|\bar{\phi}\|_{(\tau)}^2$

Therefore, $V(t)\le M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)}$ for any $t_0-\tau\le t\le t^*$

Let $t^{**}=\max\left\{t\in\left[t_0,t^*\right)\middle| V(t)\le\lambda_2\|\bar{\phi}\|_{(\tau)}^2\right\}$ which implies that:

When $t^{**}\le t\le t^*$, $V(t)\ge\lambda_2\|\bar{\phi}\|_{(\tau)}^2$ always hold. Moreover, combined with (10), it's easy to see that:

$$V(t+s)\le M\|\bar{\phi}\|_{(\tau)}^2 e^{-\lambda(t_1-t_0)} \le q\lambda_2\|\bar{\phi}\|_{(\tau)}^2 \le qV(t) \quad s\in[-\tau,0] \tag{15}$$

Which results to $\dfrac{dE\{V(t)\}}{dt}=E\{LV(t)\}\le 0$ under the condition (1).

Thus $E\left\{V\left(t^{**}\right)\right\} \geq E\left\{V\left(t^{*}\right)\right\}$ is approached according to the monotonicity in the interval $\left[t^{**}, t^{*}\right]$, further, we have $\lambda_2 \left\|\bar{\phi}\right\|_{(\tau)}^2 \geq M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_1 - t_0)}$, which contradicts (10) obviously.

Hence, we can conclude that (12) is true and (11) holds when $k = 1$.

In the second step, we suppose that the claim (11) is true for any $k = 1, 2, \cdots, m$, i.e.,

$$V(t) \leq M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_k - t_0)} \qquad t \in \left[t_{k-1}, t_k\right), \quad k = 1, 2, \cdots, m \tag{16}$$

Then, we will show that (11) also holds for $k = m + 1$, i.e.,

$$V(t) \leq M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} \qquad t \in \left[t_m, t_{m+1}\right) \tag{17}$$

We can suppose (17) is not true, which implies that there exist $\bar{t} \in \left[t_m, t_{m+1}\right)$ such that $V(\bar{t}) > M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)}$, then we can define that:

$$t' = \min\left\{ t \in \left[t_m, \bar{t}\right) \middle| V(t) \geq M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} \right\}$$

Which means $V(t) < M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} \quad t \in \left[t_m, t'\right)$. So, when $t = t_m$, since $t_k - t_{k-1} \leq d$, from the conjecture (16) and condition (2), (3), we can obtain

$$\begin{aligned} V(t_m) &= \left[x(t_m^-) + J_m\left(x(t_m^-)\right)\right]^T P \left[x(t_m^-) + J_m\left(x(t_m^-)\right)\right] \\ &\leq \beta V(t_m^-) \leq \beta M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_m - t_0)} = e^{\ln\beta} M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_m - t_0)} e^{\lambda(d-d)} \\ &\leq e^{\ln\beta} M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_m - t_0)} e^{\lambda d} e^{-\lambda(t_{m+1} - t_m)} < M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} e^{-\lambda d} \end{aligned} \tag{18}$$

Namely, $$V(t_m) < M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} e^{-\lambda d} < M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} \leq V(t') \tag{19}$$

Let $t'' = \max\left\{ t \in \left[t_m, t'\right) \middle| V(t) \leq M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} e^{-\lambda d} \right\}$, from the definition of t' and t'', it's obvious to see that, when $t \in \left[t'', t'\right]$:

$$M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} e^{-\lambda d} \leq V(t) \leq M \left\|\bar{\phi}\right\|_{(\tau)}^2 e^{-\lambda(t_{m+1} - t_0)} \tag{20}$$

Since $t \in \left[t'', t'\right]$, from the condition (2), we have $t + s \in \left[t_{m-l}, t'\right]$ for $s \in [-\tau, 0]$. Based on the (16), one observes that:

$$\begin{aligned} V(t+s) &\le M\left\|\bar{\phi}\right\|_{(\tau)}^{2} e^{-\lambda(t_{m-l+1}-t_0)} \\ &\le M\left\|\bar{\phi}\right\|_{(\tau)}^{2} e^{-\lambda(t_{m+1}-t_0)} e^{\lambda\left[(t_{m+1}-t_m)+(t_m-t_{m-1})+\cdots+(t_{m-l+2}-t_{m-l+1})\right]} \\ &\le e^{\lambda\cdot(l+1)\cdot d} M\left\|\bar{\phi}\right\|_{(\tau)}^{2} e^{-\lambda(t_{m+1}-t_0)} e^{-\lambda d} \le qV(t) \end{aligned} \tag{21}$$

Where $e^{\lambda\cdot(l+1)\cdot d} \le q$. Furthermore, $E\left\{\dot{V}(t)\right\} \le 0 \quad t \in [t'', t']$ followed from the condition (1). Then we have $E\left\{V(t'')\right\} \ge E\left\{V(t')\right\}$, moreover, lead to

$M\left\|\bar{\phi}\right\|_{(\tau)}^{2} e^{-\lambda(t_{m+1}-t_0)} e^{-\lambda d} \ge M\left\|\bar{\phi}\right\|_{(\tau)}^{2} e^{-\lambda(t_{m+1}-t_0)}$, which is a contradiction.

Here, we can draw a conclusion that, (17) is true and (16) is true for $k = m+1$. Thus, our goal (11) is reached.

By the virtue of $\lambda_1 \|x\|^2 \le V(t) \le \lambda_2 \|x\|^2$ and (11), we can further yield:

$$\|x\| \le \sqrt{\frac{M}{\lambda_1}} \left\|\bar{\phi}\right\|_{(\tau)} e^{-\frac{\lambda}{2}(t_k-t_0)} = M^* \left\|\bar{\phi}\right\|_{(\tau)} e^{-\lambda^*(t_k-t_0)} \qquad t \in [t_{k-1}, t_k) \ \ k = 1, 2, \cdots$$

Finally, our results are proved.

Remark 1. The theorem 1 presents some sufficient conditions ensuring the global exponential stability of the system concerned. It's noted that, the condition (1) cannot even guarantee the asymptotic stability of the differential system. Compared with the reference ([9]), in which X. Z. Liu employed the Razumikhin theorem as we do, our results is less conservative. In the condition (2), the minimal length of impulse interval, namely, $[t_{k-1}, t_k)$ is not limited to the time delay τ , but $t_k - t_{k-1} \ge \tau / l$ with some $l \ge 1$. The condition (3) is the constraint on the impulse effects, i.e., to guarantee the stabilizing property of impulses. We also can use this to estimate the feasible upper bounds of impulses.

4 Numerical Examples

In this section, we present a numerical example to illustrate the validity and effectiveness of the theoretical results, and a simulation graph is also drawn.

Example 1. Consider the impulsive DHNN containing two neurons:

$$\begin{cases} dx(t) = \left[Cx + Af\left(x(t)\right) + Bg\left(x(t-\tau)\right)\right]dt + \sigma\left(t, x(t), x(t-\tau)\right)d\omega(t) t \in [t_{k-1}, t_k) \\ x(t) = x\left(t_k^-\right) + J_k\left(x\left(t_k^-\right)\right) & t = t_k \\ x(s) = \phi(s) & s \in [t_0 - \tau, t_0] \end{cases}$$

Where we set

$$C=\begin{bmatrix}-0.4 & 0\\ 0 & -0.3\end{bmatrix},\ A=\begin{bmatrix}0.04 & 0.06\\ 0.01 & 0.01\end{bmatrix},\ B=\begin{bmatrix}0.01 & 0.02\\ 0.03 & 0.03\end{bmatrix},\ J=\begin{bmatrix}-1.5 & 0\\ 0 & -1.5\end{bmatrix}$$

We assume that $\tau = 0.001$, that means every 0.001s, there will be an impulse occur at that moment. The activation functions, i.e., $f(\cdot)$ and $g(\cdot)$ are assumed to be $f(x) = g(x) = \tanh(x)$, with the $L_f = L_g = I$.

Using the Matlab LMI Toolbox, we can check if all the conditions of the theorem 1 in this paper are satisfied. In example 1, with all conditions hold, we can show that the system in this example converges to the zero exponentially by the simulation graph (see figure 1 and figure 2).

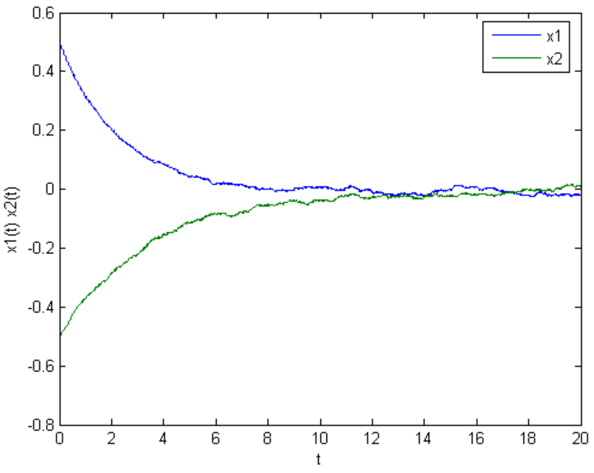


Fig. 1. The corresponding impulse-free system is not divergent

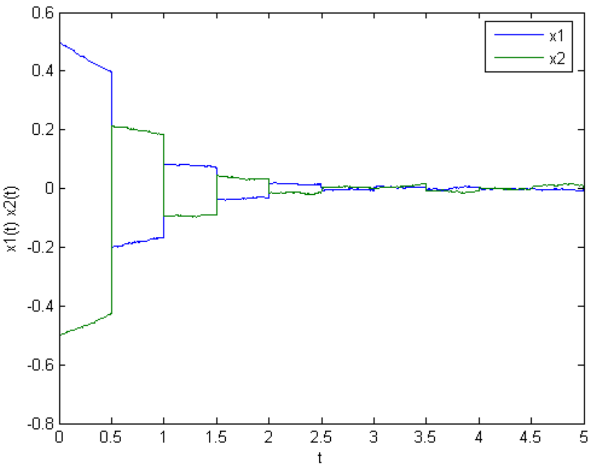


Fig. 2. Impulsive DHNN converges to the zero exponentially

5 Conclusions

Unlike the most existing papers dealing with the stability of the DHNN, we investigate the global exponential stability of the impulsive DHNN with stochastic perturbations, by means of Lyapunov-Razumikhin theorems instead of Lyapunov functional or Hanalay inequality. Some sufficient conditions guaranteeing the global exponential stability are obtained, which characterize the complicated effects of the impulses or can be used to estimate the feasible upper bound of the impulses.

References

1. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. 79, 2554–2558 (1982)
2. Hopfield, J.J.: Neurons with graded response have collective computational properties like those of two-state neurons. Proc. Natl. Acad. Sci. 81, 3088–3092 (1984)
3. Wang, C., Hill, D.J.: Deterministic learning and rapid dynamical pattern recognition. IEEE Trans. Neural Networks 18(3), 617–630 (2007)
4. Wallis, G.: Stability criteria for unsupervised temporal association networks. IEEE Trans. Neural Networks 16(2), 301–311 (2005)
5. Cao, J., Xiao, M.: Stability and Hopf bifurcation in a simplified BAM neural network with two time delays. IEEE Trans. Neural Networks 18(2), 416–430 (2007)
6. Anokhin, A., Berezansky, L., Braverman, E.: Exponential stability of linear delay impulsive differential equations. J. Math. Anal. Appl. 193, 923–941 (1995)
7. Li, C., Shen, Y.Y., Feng, G.: Stabilizing effects of impulse in delayed BAM neural networks. IEEE Transactions on Circuits and Systems-II 53(12), 1284–1288 (2008)
8. Liu, X., Wang, Q.: The method of Lyapunov functional and exponential stability of impulsive systems with time delay. Nonlinear Anal. 66, 1465–1484 (2007)
9. Liu, X., Wang, Q.: Impulsive stabilization of high-order Hopfield-Type neural networks with time-varying delays. IEEE Trans. Neural Networks 19(1) (2008)
10. Feng, W., Yang, S.X., Fu, W., Wu, H.: Robust stability analysis of uncertain stochastic neural networks with interval time-varying delay. Chaos, Solitons and Fractals 41, 414–424 (2009)
11. Luo, M., Zhong, S., Wang, R., Kang, W.: Robust stability analysis for discrete-time stochastic neural networks systems with time-varying delays. Appl. Mathematics and Computation 209, 305–313 (2009)
12. Han, J., Qiu, J., Jia, H., Meng, H.: Robust Stability for Stochastic Interval Hopfield Neural Networks with Time Delays. In: 2010 Sixth International Conference on Natural Computation, vol. 2, pp. 801–805. IEEE Press, Los Alamitos (2010)
13. Mao, X.: Razumikhin type theorems on exponential stability of stochastic functional differential equations. Stochastic Processes and their Applications 65(2), 233–250 (1996)
14. Yang, Z., Zhu, E., Xu, Y., Tan, Y.: Razumikhin-type theorems on exponential stability of stochastic functional differential equations with infinite delay. Acta. Appl. Math., 1–13 (2010)
15. Morita, M.: Associative memory with non-monotone dynamics. Neural Networks 6, 115–126 (1993)
16. Yoshizawa, S., Morita, M., Amari, S.I.: Capacity of associative memory using a nonmonotonic neuron model. Neural Networks 6, 167–176 (1993)

New LMI-Based Criteria for Global Robust Stability of Cohen-Grossberg Neural Networks with Time-Varying Delays

Chaojin Fu, Dahu Li, and Huan Tong

College of Mathematics and Statistics, Hubei Normal University, Huangshi 435000, China
{chaojinfu,dahuli1986}@126.com

Abstract. In this paper, some sufficient conditions for global robust exponential stability of neural networks with time-varying delays are presented. On basis of the obtained results, some linear matrix inequality(LMI)criteria are derived. A comparison of the present criteria with the previous criteria is made. Moreover, an example is given to show the effectiveness of the obtained results.

Keywords: Neural networks, Global robust stability, Time-varying delays.

1 Introduction

Cohen and Grossberg proposed a class of neural networks in 1983 [1], which include Hopfield neural networks [2], this model has received increasing interest due to its promising potential for applications in classification, parallel computation, associative memory, especially in solving some optimization problems. Such applications rely on the qualitative properties of stability. Thus, the qualitative analysis of these dynamic behaviors is a prerequisite step for the practical design and application of neural networks. Because of the finite speed of switching and transmission of signals in a network, time delays are inevitably resent in electronic implementation of neural networks, which may influence the stability of the entire network by creating oscillatory or unstable phenomena. Therefore, the study of stability of neural networks with delay is practically required [3-12]. Recently, study on the importance of delays in BAM Cohen-Grossberg neural networks have been investigated in [8][9]. and some authors [10-12] analyze the global exponential stability of the Cohen-Grossberg neural networks. Global robust stability analysis of the Cohen-Grossberg neural networks had been proposed in [3-5]. The methods most of them used is differential inequality technique and Lyapunov direct method and LMI. In this work, By introducing a new Lyapunov-Krasovskii functional and considering some useful terms when estimating the upper bound of the derivative of Lyapunov functional, new robust exponential stability criteria are established in term of linear matrix inequality(LMI).

© Springer-Verlag Berlin Heidelberg 2011

The remainder of this paper is organized as follows. In section 2, model description and preliminaries are given. And then in section 3, our main results are presented. In section 4, a numerical example is supplied to illustrate the effectiveness of our obtained results. Finally, in section 5, our conclusion is given.

2 Model Description and Preliminaries

Cohen-Grossberg neural networks model with time-varying delays is described by the following differential equation:

$$\begin{aligned}\dot{x}_i(t) = &- d_i(x_i(t))[-c_i(x_i(t)) + \sum_{j=1}^{n} a_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} b_{ij} f_j(x_j(t-\tau_j(t))) + I_i],\\ &i = 1, \cdots, n, \qquad (1)\end{aligned}$$

where n denotes the number of neurons; $x_i(t)$ is the state of the ith neuron at time t, the state vector $x(t) := (x_1(t), x_2(t), \cdots, x_n(t))^T$; $d_i(x_i)$ represent an amplification function, and $c_i(x_i)$ is a behaved function. The functions $f_j(\cdot)$ represent the input-output activations, $f := (f_1, \cdots, f_n)^T$; $A := (a_{ij}) \in R^{n\times n}$ and $B := (b_{ij}) \in R^{n\times n}$ are the interconnection weight matrix and the delayed interconnection weight matrix, respectively; $\tau_j(t)$ denotes the time delay associated with the ith neuron; I_i is the external bias on the ith neuron, $I := (I_1, \cdots, I_n)^T$. The quantities d_i,c_i, a_{ij}, b_{ij} can be intervalised as follows:

$$\begin{aligned}
&D_I := \{D = diag(d_i) : 0 \prec \underline{D} \preceq D \preceq \overline{D}, i.e., 0 < \underline{d_i} \le d_i \le \overline{d}_i, i = 1, 2, \cdots, n\}\\
&C_I := \{C = diag(c_i) : 0 \prec \underline{C} \preceq C \preceq \overline{C}, i.e., 0 < \underline{c_i} \le c_i \le \overline{c}_i, i = 1, 2, \cdots, n\}\\
&A_I := \{A = diag(a_i) : \underline{A} \preceq A \preceq \overline{A}, i.e., \underline{a_i} \le a_i \le \overline{a}_i, i = 1, 2, \cdots, n\}\\
&B_I := \{B = diag(b_i) : \underline{B} \preceq B \preceq \overline{B}, i.e., \underline{b_i} \le b_i \le \overline{b}_i, i = 1, 2, \cdots, n\} \qquad (2)
\end{aligned}$$

where $\underline{A} \le A$ implies that the elements of matrices $\underline{A}$,A satisfy the inequality $\underline{a_{ij}} \le a_{ij}$.

System (1) is supplemented with initial conditions of the following form:$x_i(\theta) = \phi_i(\theta), \theta \in [-\tau, 0], (i = 1, 2, \cdots, n)$, where $\phi_i(\theta)$ is continuous for $\theta \in [-\tau, 0]$.

For convenience, let $\nu = (\nu_1, \nu_2, \cdots, \nu_n) \in R^n$ be a column vector and $Q = (q_{ij})_{n\times n}$ be a real matrix. The three commonly used vector norms $\|\nu\|_1, \|\nu\|_2$, $\|\nu\|_\infty$ are defined as :

$$\|\nu\|_1 = \sum_{i=1}^{n} |\nu_i|\,, \quad \|\nu\|_2 = (\sum_{i=1}^{n} \nu_i^2)^{\frac{1}{2}}\,, \quad \|\nu\|_\infty = \max_{1\le i\le n} |\nu_i|$$

The three commonly used matrix norms $\|Q\|_1$, $\|Q\|_2$ and $\|Q\|_\infty$ are defined as follows:

$$\|Q\|_1 = \max_{1\le i\le n} \sum_{j=1}^{n} |q_{ji}|\,, \quad \|Q\|_2 = [\Lambda_{max}(Q^TQ)]^{\frac{1}{2}}, \quad \|Q\|_\infty = \max_{1\le i\le n} \sum_{j=1}^{n} |q_{ij}|$$

Furthermore, we introduce the following assumptions on (1):

(A$_1$) $f_i(x_i)(i = 1, 2, \cdots, n) : R \to R$ are Lipschitz continuous and monotonically nondecreasing, that is, there exist constants $\mu_i > 0$ such that

$$0 \le f_i(x) - f_i(y) \le \mu_i \left| x - y \right|$$

(A$_2$) $\tau_i(t)(i = 1, 2, \cdots, n)$ are bounded differential functions of time t, and the following conditions are satisfied:

$$0 \le \tau_i(t) \le \tau, \quad 0 \le \dot{\tau}_i(t) \le h_i < 1$$

Denote $\Lambda = diag(\mu_1, \mu_2, \cdots, \mu_n), \mu_M = \max\limits_{1 \le i \le n}(\mu_i)$ and $h = \max\limits_{1 \le i \le n}(h_i)$.

(A$_3$) there exist constants $\overline{\alpha}_i > 0, \underline{\alpha}_i > 0$ such that $0 < \underline{\alpha}_i \le \alpha_i(x_i) \le \overline{\alpha}_i > 0$.

Definition 1. [5]The neural network defined by (1) with the parameter ranges defined by (2) is globally exponentially robust stable if system (1) has a unique equilibrium point $x^* = (x_1^*, x_2^*, \cdots, x_n^*)^T$ for all $D \in D_I, C \in C_I, A \in A_I$ and $B \in B_I$. and there exist constants $\zeta > 0$ and $\alpha \ge 1$ such that

$$\|x(t) - x^*\| \le \alpha \|\phi(\theta) - x^*\| \exp\{-\zeta T\}, \forall t > 0$$

Lemma 1. [5]For $B \in [\overline{B}, \underline{B}]$, the following inequalities hold:

$$\|B\|_2 \le \|B^*\|_2 + \|B_*\|_2$$

where $B^* = \frac{1}{2}(\overline{B} + \underline{B}), B_* = \frac{1}{2}(\overline{B} - \underline{B})$.

Lemma 2. [10]Continuous map $H(x)$:$R^n \to R^n$ is homeomorphic if:
(1)$H(x)$ is injective;
(2) $\lim\limits_{\|x\| \to \infty} \|H(x)\| = \infty$.

Lemma 3. [6]The following LMI:

$$\begin{pmatrix} Q(x) & S(x) \\ S^T(x) & R(x) \end{pmatrix} > 0$$

where $Q(x) = Q^T(x), R(x) = R^T(x) and S(x)$ depend affinely on x, is equivalent to

$$R(x) > 0 \quad and \quad Q(x) - S(x)R^{-1}(x)S^T(x) > 0$$

Lemma 4. Let $\varepsilon > 0$, for any $x, y \in R^n$ and matrix A, then

$$x^T A y \le \frac{1}{2\varepsilon} x^T A A^T x + \frac{\varepsilon}{2} y^T y$$

Lemma 5. $B \in B_I$, then, the following inequality holds:

$$\|B\|_2^2 \le \|B^*\|_2^2 + \|B_*\|_2^2 + 2\|B_*^T \left| B^* \right| \|_2$$

where $B^* = \frac{1}{2}(\overline{B} + \underline{B}), B_* = \frac{1}{2}(\overline{B} - \underline{B})$.

Proof. For any vector $x = (x_1, x_2, \cdots, x_n) \in R^n$, we have

$$\begin{aligned} x^T B^T B x &\leq x^T \overline{B}^T \overline{B} x \\ &= x^T (B^* + B_*)^T (B^* + B_*) x \\ &= x^T {B^*}^T B^* x + 2x^T B_*^T B^* x + x^T {B_*}^T B_* x \\ &\leq \|B^*\|_2^2 \|x\|_2^2 + 2\| {B_*}^T \, |B^*| \, \|_2 \|x\|_2^2 + \|B_*\|_2^2 \|x\|_2^2 \\ &= (\|B^*\|_2^2 + 2\| {B_*}^T \, |B^*| \, \|_2 + \|B_*\|_2^2) \|x\|_2^2 \end{aligned}$$

implying that

$$\|B\|_2^2 \leq \|B^*\|_2^2 + \|B_*\|_2^2 + 2\|B_*^T \, |B^*| \, \|_2$$

3 Main Results

Theorem 1. If (A$_1$), (A$_2$) hold, the neural network model (1) has a unique equilibrium point if there are positive diagonal matrix $P = diag(p_1, p_2, \cdots, p_n)$ and a constant ε such that

$$\Omega_1 = 2P\underline{C}\Lambda^{-1} + S - [\varepsilon + \frac{1}{\varepsilon}\|P\|_2^2(\|B^*\|_2^2 + \|B_*\|_2^2 + 2\|B_*^T \, |B^*| \, \|_2)]I > 0$$

where $S = (s_{ij})_{n\times n}$ with $s_{ii} = -2p_i\overline{a}_{ii}$, $s_{ij} = -max(|p_i\overline{a}_{ij} + p_j\overline{a}_{ji}|, |p_i\underline{a}_{ij} + p_j\underline{a}_{ji}|)$ for $i \neq j$.

Proof. Let$x^* := (x_1^*, \cdots, x_n^*)^T$ denote an equilibrium point of system (1). Then x^*satisfies

$$D(x^*)[-C(x^*) + Af(x^*) + Bf(x^*) + I] = 0 \tag{3}$$

Since $D(x^*)$ is a positive diagonal matrix, (3) is equivalent to the following equation:

$$-C(x^*) + Af(x^*) + Bf(x^*) + I = 0$$

Let

$$H(x) = -Cx + Af(x) + Bf(x) + I \tag{4}$$

To complete the proof, it is sufficient to show that $H(x)$ is a homeomorphism on R^n. Based on Lemma 2, we first prove that the map $H(x)$ is injective on R^n. For two vectors $x, y \in R^n, x \neq y$, it can be obtained that

$$H(x) - H(y) = -C(x-y) + A(f(x) - f(y)) + B(f(x) - f(y)) \tag{5}$$

If $f(x) - f(y) = 0$, then $H(x) - H(y) = -C(x-y)$, it is clear that $H(x) \neq H(y)$ if $x \neq y$. If $f(x) - f(y) \neq 0$, multiplying both sides of by $2(f(x) - f(y))^T P$, we have

$$\begin{aligned} 2(f(x) - f(y))^T P(H(x) - H(y) = &- 2(f(x) - f(y))^T PC(x-y) \\ &+ 2(f(x) - f(y))^T PA(f(x) - f(y)) \\ &+ 2(f(x) - f(y))^T PB(f(x) - f(y)) \end{aligned} \tag{6}$$

Considering the three parts of the right side of Eq.(6) separately, we first have

$$\begin{aligned}-2(f(x)-f(y))^T PC(x-y) \le & -2(f(x)-f(y))^T PC\Lambda^{-1}(f(x)-f(y)) \\ \le & -2(f(x)-f(y))^T P\underline{C}\Lambda^{-1}(f(x)-f(y)) \end{aligned} \quad (7)$$

We also have

$$\begin{aligned}& 2(f(x)-f(y))^T PA(f(x)-f(y)) \\ =&(f(x)-f(y))^T(PA+A^TP)(f(x)-f(y)) \\ =&\sum_{i=1}^{n}\sum_{j=1}^{n}(p_i a_{ij}+p_j a_{ji})(f_i(x_i)-f_i(y_i))(f_j(x_j)-f_j(y_j)) \\ \le&\sum_{i=1}^{n}2p_i\overline{a}_{ii}(f_i(x_i)-f_i(y_i))^2 \\ +&\sum_{i=1}^{n}\sum_{j=1,j\ne i}^{n}|p_i a_{ij}+p_j a_{ji}|\,|(f_i(x_i)-f_i(y_i))(f_j(x_j)-f_j(y_j))| \\ \le&-\sum_{i=1}^{n}s_{ii}(f_i(x_i)-f_i(y_i))^2 \\ -&\sum_{i=1}^{n}\sum_{j=1,j\ne i}^{n}s_{ij}\,|(f_i(x_i)-f_i(y_i))(f_j(x_j)-f_j(y_j))| \\ =&-\left|(f(x)-f(y))^T\right| S\,|f(x)-f(y)| \end{aligned} \quad (8)$$

Moreover, it can be derived by Lemma 4 and Lemma 5 that

$$\begin{aligned}2(f(x)-f(y))^T PB(f(x)-f(y)) \le&\frac{1}{\varepsilon}(f(x)-f(y))^T PB(PB)^T(f(x)-f(y)) \\ &+\varepsilon(f(x)-f(y))^T(f(x)-f(y)) \\ \le&\frac{1}{\varepsilon}(f(x)-f(y))^T\|P\|_2^2\|B\|_2^2(f(x)-f(y)) \\ &+\varepsilon(f(x)-f(y))^T(f(x)-f(y)) \\ \le&\frac{1}{\varepsilon}(f(x)-f(y))^T\|P\|_2^2(\|B^*\|_2^2+\|B_*\|_2^2 \\ &+2\|B_*^T\,|B^*|\,\|_2)(f(x)-f(y)) \\ &+\varepsilon(f(x)-f(y))^T(f(x)-f(y)) \end{aligned} \quad (9)$$

Substing(7), (8) and (9) into (6), yields

$$2(f(x)-f(y))^T P(H(x)-H(y) \le -\left|(f(x)-f(y))^T\right| \Omega_1\,|f(x)-f(y)| \quad (10)$$

Since $\Omega_1>0$, it follows that

$$2(f(x)-f(y))^T P(H(x)-H(y)<0, \forall f(x)\ne f(y)$$

Therefore, $H(x) \neq H(y)$ for all $x \neq y$, i.e., $H(x)$ is injective on R^n.

Second, we will show that $\|H(x)\| \to \infty$, when $\|x\| \to \infty$. If $y = 0$, it turns out that

$$\begin{aligned}2(f(x)-f(0))^T P(H(x)-H(0) &\leq -\left|(f(x)-f(0))^T\right| \Omega_1 \left|f(x)-f(0)\right| \\ &\leq -\lambda_{min}(\Omega_1)\left|(f(x)-f(0))^T(f(x)-f(0))\right|\end{aligned}$$

It follows that

$$2(f(x)-f(0))^T P(H(x)-H(0) \geq \lambda_{min}(\Omega_1)\|f(x)-f(0)\|_2^2$$

Furthermore,

$$\|H(x)\|_2 + \|H(0)\|_2 \geq \frac{\lambda_{min}(\Omega_1)}{2\|P\|_2}(\|f(x)\|_2 - \|f(0)\|_2)$$

Since $\|H(0)\|_2$ and $\|f(0)\|_2$ are finite, it is obvious that $\|H(x)\|_2 \to +\infty$ as $\|f(x)\|_2 \to +\infty$. On the other hand, for unbounded activation functions, $\|f(x)\|_2 \to +\infty$ implies $\|x\|_2 \to +\infty$; for bounded activation functions, it can be obtained from (4) that $\|H(x)\|_2 \to +\infty$ as $\|x\|_2 \to +\infty$. From Lemma 2, $H(x)$ is a homeomorphism on R^n. The proof is completed.

Let (1) has an unique equilibrium point $x^* := (x_1^*, x_2^*, \cdots, x_n^*)^T$, we make a transformation: $y_i(t) := x_i(t) - x_i^*$, $\alpha_i(y_i(t)) = d_i(y_i(t) + x_i^*)$, $\beta_i(y_i(t)) = c_i(y_i(t) + x_i^*) - c_i(x_i^*) \geq \gamma_i(y_i(t))$, $g_i(y_i(t)) := f_i(y_i(t) + x_i^*) - f_i(x_i^*)$, $\Phi_i(t) = \phi_i(t) - x_i^*$, $i = 1, 2, \cdots, n$, then system (1) is transformed into the following system:

$$\begin{aligned}\dot{y}_i(t) =& \alpha_i(y_i(t))[-\beta_i(y_i(t)) + \sum_{j=1}^{n} a_{ij} g_j(y_j(t)) + \sum_{j=1}^{n} b_{ij} g_j(y_j(t-\tau_j(t)))], \\ & i = 1, \cdots, n,\end{aligned} \tag{11}$$

by the initial conditions of system (1), we obtain (11) is supplemented with initial conditions $y_i(t) = \Phi_i(t) \in C((-\infty, t_0]; R)$. $\Phi(t) := (\Phi_1(t), \Phi_2(t), \cdots, \Phi_n(t))^T$, $\|\Phi\| := \sup \|\Phi(t_0+\theta)\|_\infty < +\infty$, $\theta \in (-\infty, 0]$. Obviously, g_j are Lipschitz continuous, and g_j also satisfy (A$_1$), $j = 1, 2, \cdots, n$. By this way, we shift the equilibrium point x^* of system (1) to the origin of system (11).

Theorem 2. If (A$_1$), (A$_2$),(A$_3$) satisfied, then system (1) has an unique equilibrium point, which is global exponentially robust stable, if there exist positive diagonal matrices $P = diag(p_1, p_2, \cdots, p_n)$ and a constant ε such that

$$\Omega_2 = 2P\gamma\Lambda^{-1} + S - [\varepsilon + \frac{1}{(1-h)\varepsilon}\|P\|_2^2(\|B^*\|_2^2 + \|B_*\|_2^2 + 2\|B_*^T\,|B^*|\,\|_2)]I > 0$$

where $\gamma \leq \underline{C}$.

Proof. It is clear that $0 < \Omega_2 < \Omega_1$. Consider the following Lyapunov functional:

$$\begin{aligned} V(y(t)) =& e^{bt} y^T(t) y(t) + 2e^{bt} \sum_{i=1}^{n} p_i \int_0^{y_i(t)} \frac{g_i(s)}{\alpha_i(s)} \mathrm{d}s \\ &+ \varepsilon e^{b\theta} \theta \sum_{i=1}^{n} \int_{t-\tau_i(t)}^{t} g_i^T(y_i(\theta)) g_i(y_i(\theta)) \mathrm{d}\theta \end{aligned} \tag{12}$$

It is obvious that

$$V(y(t)) \geq e^{bt} y^T(t) y(t) \tag{13}$$

Calculating the derivative of $V_(y(t))$ along the solution of system (11) yields

$$\begin{aligned} \dot{V}(y(t)) =& b e^{bt} [y^T(t) y(t) + 2e^{bt} \sum_{i=1}^{n} p_i \int_0^{y_i(t)} \frac{\mu_i s}{\underline{\alpha}_i} \mathrm{d}s] \\ &+ 2e^{bt} y^T(t) \alpha(y(t)(-\beta(y(t)) + Ag(y(t)) + Bg(y(t-\tau(t)))) \\ &+ 2e^{bt} P g^T(y(t))(-\beta(y(t)) + Ag(y(t)) + Bg(y(t-\tau(t)))) \\ &+ \varepsilon e^{bt} \sum_{i=1}^{n} g_i^T(y_i(t)) g_i(y_i(t)) \\ &- \varepsilon e^{b(t-\tau_i(t))} \sum_{i=1}^{n} (1-\dot{\tau}_i(t)) g_i^T(y_i(t-\tau_i(t))) g_i(y_i(t-\tau_i(t))) \\ \leq& b e^{bt} [y^T(t) y(t) + 2e^{bt} \sum_{i=1}^{n} p_i \int_0^{y_i(t)} \frac{g_i(s)}{\underline{\alpha}_i(s)} \mathrm{d}s] \\ &- 2e^{bt} \underline{\alpha} \gamma \left| y^T(t) \right| |y(t)| + e^{bt} (\overline{\alpha} A + \overline{\alpha} A^T) \left| y^T(t) \right| |g(y(t))| \\ &+ e^{bt} (\overline{\alpha} B + \overline{\alpha} B^T) \left| y^T(t) \right| |g(y(t-\tau(t)))| \\ &- 2e^{bt} P \gamma \Lambda^{-1} \left| g^T(y(t)) \right| |g(y(t))| - e^{bt} \left| g^T(y(t)) \right| S |g(y(t))| \\ &+ e^{bt} (PB + B^T P) |g(y(t)| |g(y(t-\tau(t)))| + e^{bt} \varepsilon \left| g^T(y(t)) \right| |g(y(t))| \\ &- e^{bt} \varepsilon (1-h) \left| g^T(y(t-\tau(t))) \right| |g(y(t-\tau(t)))| \end{aligned} \tag{14}$$

and, we can write (12) as following

$$\dot{V}(y(t)) \leq b e^{bt} y^T(t) y(t) + e^{bt} \begin{pmatrix} |y(t)| \\ |g(y(t))| \\ |g(y(t-\tau(t)))| \end{pmatrix}^T \Sigma \begin{pmatrix} |y(t)| \\ |g(y(t))| \\ |g(y(t-\tau(t)))| \end{pmatrix} \tag{15}$$

where

$$\Sigma = \begin{pmatrix} -2\underline{\alpha}\gamma & \overline{\alpha} A^T & \overline{\alpha} B^T \\ \overline{\alpha} A & -2P\gamma \Lambda^{-1} - S + \varepsilon & PB \\ \overline{\alpha} B & B^T P & -\varepsilon(1-h) \end{pmatrix}$$

From Lemma 4 , $\Sigma < 0$ is equivalent to

$$\begin{aligned}&2P\gamma\Lambda^{-1} + S - [\varepsilon + \frac{1}{(1-h)\varepsilon}PBB^TP]\\ \geq&2P\gamma\Lambda^{-1} + S - [\varepsilon + \frac{1}{(1-h)\varepsilon}\|P\|_2^2(\|B^*\|_2^2 + \|B_*\|_2^2 + 2\|B_*^T\,|B^*|\,\|_2)]I\\ >&0\end{aligned}$$

Let $0 < b < \lambda_{min}(-\Sigma)$, then

$$\dot{V}(y(t)) \leq \lambda_{min}(-\Sigma)e^{bt}y^T(t)y(t) + e^{bt}\begin{pmatrix}|y(t)|\\|g(y(t))|\\|g(y(t-\tau(t)))|\end{pmatrix}^T\Sigma\begin{pmatrix}|y(t)|\\|g(y(t))|\\|g(y(t-\tau(t)))|\end{pmatrix} \tag{16}$$

So

$$\dot{V}(y(t)) \leq 0$$

As a result,$V(y(t)) \leq V(y(0))$ for all $t \geq 0$.

On the other hand,

$$\begin{aligned}V(y(0)) =& y^T(t)y(t) + 2\sum_{i=1}^{n}p_i\int_0^{y_i(0)}\frac{g_i(s)}{\alpha_i(s)}\mathrm{d}s + \varepsilon\sum_{i=1}^{n}\int_{-\tau_i(0)}^{0}e^{b\theta}g_i^T(y_i(\theta))g_i(y_i(\theta))\mathrm{d}\theta\\ &\leq \|y(0)\|_2^2 + \max_{1\leq i\leq n}(p_i\frac{\mu_i}{\underline{\alpha}_i})\|y(0)\|_2^2 + {\mu_M}^2\int_{-\tau}^{0}e^{b\theta}\|y(\theta)\|_2^2\mathrm{d}\theta\\ &\leq [1 + \max_{1\leq i\leq n}(p_i\frac{\mu_i}{\underline{\alpha}_i}) + \frac{1-e^{-b\tau}}{b}]\sup_{-\tau<\xi<0}\|y(\theta)\|_2^2\end{aligned} \tag{17}$$

Combining (13) with (17) yields

$$e^{bt}\|y(t)\|_2^2 \leq [1 + \max_{1\leq i\leq n}(p_i\frac{\mu_i}{\underline{\alpha}_i}) + \frac{1-e^{-b\tau}}{b}]\sup_{-\tau<\xi<0}\|y(\theta)\|_2^2$$

that is

$$\|y(t)\|_2^2 \leq \alpha\sup_{-\tau<\xi<0}\|y(\theta)\|_2 e^{-bt/2}$$

where

$$\alpha = \sqrt{1 + \max_{1\leq i\leq n}(p_i\frac{\mu_i}{\underline{\alpha}_i}) + \frac{1-e^{-b\tau}}{b}} \geq 1$$

Thus,by Definition 1, it follows that the delayed neural network is globally robust exponentially stable. This completes the proof.

4 Comparison and Corollary

Corollary 1 [13]. If (A$_1$), (A$_2$) satisfied, then Hopfield neural network has an unique equilibrium point, which is global asymptotically robust stable, if there exist positive diagonal matrices $P = diag(p_1, p_2, \cdots, p_n)$ such that

$$\Omega_3 = 2P\underline{C}\Lambda^{-1} + S - H - [\|P\|_2^2\|H^{-1}\|_2(\|B^*\|_2 + \|B_*\|_2)^2]I > 0$$

where $S = (s_{ij})_{n\times n}$ with $s_{ii} = -2p_i\overline{a}_{ii}$, $s_{ij} = -max(|p_i\overline{a}_{ij} + p_j\overline{a}_{ji}|, |p_i\underline{a}_{ij} + p_j\underline{a}_{ji}|)$ for $i \neq j$.

The Corollary1 is based on the following inequality:

$$\begin{aligned} g^T(x^*)(2P\underline{C}\Lambda^{-1} - PA - A^TP - H - [\|P\|_2^2\|H^{-1}\|_2(\|B^*\|_2 + \|B_*\|_2)^2]I)g(x^*) \\ \geq g^T(x^*)(2P\underline{C}\Lambda^{-1} + S - H - [\|P\|_2^2\|H^{-1}\|_2(\|B^*\|_2 + \|B_*\|_2)^2]I)g(x^*) \end{aligned}$$

Note that the above inequality is not always correct. By Lemma 5, the obtained stability conditions are less conservative.

Corollary 2. If (A_1), (A_2), (A_3) satisfied, then system (1) has an unique equilibrium point, which is global exponentially robust stable, if there exist positive diagonal matrices $P = diag(p_1, p_2, \cdots, p_n)$ and a constant ε such that

$$\Omega_4 = 2P\underline{C}\Lambda^{-1} + PA + A^TP - [\varepsilon + \frac{1}{\varepsilon}\|P\|_2^2\|B\|_2^2]I > 0$$

Corollary 3. If (A_1), (A_2), (A_3) satisfied, then system (1) has an unique equilibrium point, which is global exponentially robust stable, if there exist positive diagonal matrices $P = diag(p_1, p_2, \cdots, p_n)$ and a constant ε such that

$$\Omega_5 = 2P\underline{C}\Lambda^{-1} + PA + A^TP - [\varepsilon + \frac{1}{\varepsilon}\|P\|_2^2(\|B^*\|_2 + \|B_*\|_2)^2]I > 0$$

5 Illustrative Example

Consider the neural system (1) with the following network parameters:

$$\underline{A} = \begin{pmatrix} -1 & -1 \\ -1 & -1 \end{pmatrix}, \overline{A} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \underline{B} = \begin{pmatrix} -1 & -1 \\ -1 & -1 \end{pmatrix}, \overline{B} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \gamma\Lambda^{-1} = \begin{pmatrix} 5 & 0 \\ 0 & 5 \end{pmatrix}$$

The matrix A^*, A_*, B^*, B_*, are obtained as follows:

$$A^* = B^* = 0, A_* = \begin{pmatrix} -1 & -1 \\ -1 & -1 \end{pmatrix}, B_* = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

then

$$\Omega_2 = 2P\gamma\Lambda^{-1} + S - [\varepsilon + \frac{1}{(1-h)\varepsilon}\|P\|_2^2(\|B^*\|_2^2 + \|B_*\|_2^2 + 2\|B_*^T\,|B^*|\,\|_2)]I > 0$$

where $P = diag(p_1, p_2) = diag(1, 1)$, $\varepsilon = 1$, $h = \frac{1}{2}$,$S = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$, so by Theorem 2, the system has an unique equilibrium point, which is global exponential robust stable.

6 Conclusion

In this paper, we study the global exponential robust stability of Cohen-Grossberg neural networks with time-varying delays. We give a new method to proof Lemma 5, and derive some new sufficient conditions based on for global robust stability of delayed neural networks based on LMI and differential inequality technique. In comparison with some recent results reported in the literature, the new stability criteria is very simple to give and less conditions to prove the new stability criteria.

References

1. Cohen, M.A., Grossberg, S.: Stability and global pattern formulation and memory storage by competitive neural networks. IEEE Trans. Syst. Man Cybern. 13, 815–826 (1983)
2. Hopfield, J.: Neurons with graded response have collective computational properties like those of two state neurons. Proceedings of the National Academy of Sciences, USA 81, 3088–3092 (1984)
3. Rong, L.: LMI-based criteria for robust stability of CohenCGrossberg neural networks with delay. Physics Letters A 339, 63–73 (2005)
4. Feng, J., Xu, S.: New criteria on global robust stability of CohenCGrossberg neural networks with time-varying delays. Neurocomputing 72, 445–457 (2008)
5. Wang, Z., Zhang, H., Yu, W.: Robust stability criteria for interval CohenCGrossberg neural networks with time varying delay. Neurocomputing 72, 1105–1110 (2009)
6. Pana, J., Liu, X., Zhong, S.: Stability criteria for impulsive reaction-diffusion Cohen-Grossberg neural networks with time-varying delays. Mathematical and Computer Modelling 51, 1037–1050 (2010)
7. Zhong, S., Li, C., Liao, X.: Global stability of discrete-time CohenCGrossberg neural networks with impulses. Neurocomputing 73, 3132–3138 (2010)
8. Syed Ali, M., Balasubramaniam, P.: Robust stability of uncertain fuzzy CohenC-Grossberg BAM neural networks with time-varying delays. Expert Systems with Applications: An International Journal 7 (2006)
9. Nie, X., Cao, J.: Stability analysis for the generalized Cohen-Grossberg neural networks with inverse Lipschitz neuron activations. Computers Mathematics with Applications 57, 898–1221 (2009)
10. Chen, S., Zhao, W., Xu, Y.: New criteria for globally exponential stability of delayed Cohen-Grossberg neural network. Mathematics and Computers in Simulation 79, 1527–1543 (2009)
11. Cui, B.T., Wu, W.: Global exponential stability of CohenCGrossberg neural networks with distributed delays. Neurocomputing 72, 386–391 (2008)
12. Liao, X., Yang, J., Guo, S.: Exponential stability of CohenCGrossberg neural networks with delays. Communications in Nonlinear Science and Numerical Simulation 13, 1767–1775 (2008)
13. Cao, J., Li, H.X., Han, L.: Novel results concerning global robust stability delayed neural networks. Nonlinear Analysis: Real World Applications 7, 458–469 (2006)

Asymptotical Stability of an SEIS Epidemic Model with Latent Age Dependence and Generally Nonlinear Contact Rate*

Yeling Liu and Zhonghua Zhang**

School of Sciences, Xi'an University of Science and Technology,
Xi'an, Shaanxi 710054, China
`wwwzhonghua@sohu.com`

Abstract. In this paper, an SEIS epidemic model with latent age and generally nonlinear contact rate is formulated. The existence and asymptotic stability of equilibrium are discussed, respectively. In the same time, a general condition is obtained by the similar method utilized in [12], under which the endemic equilibrium is exponentially asymptotically stable. At last, a special example is presented to verified this condition.

Keywords: SEIS epidemic model; Latent age; Contact rate; Stability.

1 Introduction

In most epidemiological models, it has been assumed that all infected individuals have equal infectivity during their infective period, and this assumption has been proved to be reasonable in the study of the dynamics of the communicable diseases such as influenza [1] and the sexually transmitted diseases such as gonorrhea [2]. However, the early infectivity experiments [3] reported in Francis together with the measurements of HIV antigen and antibody titers have supported the possibility of an early infectivity peak (a few weeks after exposure) and a subsequent infectivity plateau (one year or so before the onset of full-blown of AIDS [4]). Therefore, there are enough reasons to study the possible effects of variable infectivity on epidemic dynamics. The initial work was owed to Kermack and Mckendrick [5]-[8], in which the infectivity was allowed to depend on infection-age (*namely*, the time that has evolved beginning with the moment of infection). However, the epidemic models with infection-age were largely neglected until 1970s [9]-[10]. Recently, this kind of models have been considered extensively (*see* [11]-[16] and the references therein for examples).

* *2000 Mathematics Subject Classifications*: 34D20; 34D23; 45D05; 44A10. This work was supported by Mathematics Tianyuan Funds of NSFC (No:11026133);Scientific Research Plan Projects of Shaanxi Education Department (No.09JK601), National Science Foundation for Post-doctoral Scientists of China (No.20090461281), Training Fund of Xi'an University of Science and Technology under the contract 200836 and the Dr. Start-up Fund of Xi'an University of Science and Technology.

** Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

In [11], M. Y. Kim and F. A. Milner formulated an SIR epidemic model with screening together with variable infectivity and proved the global existence and uniqueness of the positive solution. Subsequently, M. Y. Kim discussed the asymptotical properties of equilibria in [12]. In [13], to investigate the role of variable infectivity and variable incubation period on the dynamics of HIV transmission in a homogeneously mixed population, and discussed both the asymptotical behavior of the equilibria and the possibility whether the undamped oscillations occur or not. In [14], C. M. Kribs-Zaleta and M. Martcheva modeled a disease with acute and chronic infective stages, variable infectivity and recovery rates and exhibited backward bifurcations under some conditions. In [15], H. Inaba and H. Sekine studied the stability of the equilibria of an infection-age-dependent model for chagas disease. In [16], Jia Li, *etc.* considered epidemiological models for the transmission of a pathogen that can mutate in the host to create a second infectious mutant strain and showed that there exists a Hopf bifurcation where the endemic equilibrium loses its stability under certain circumstances.

The latent period of epidemic disease is the lag time between the exposure to a disease-causing agent and the onset of disease caused by the agent. Some sexually transmitted diseases such as Aids, Condyloma Acuminata, *et al.* possess long latent period, which plays important role on the spreading of the diseases. It is because of that the infected individual will restrict his or her behaviors once he or she is diagnosed to be infected by virus, pathogen, *et al.* Therefore, latent age as an necessary factor should be considered in process of modeling. On the other hand, with the development of social economy and communication, the contact among humans become more and more complex. It is difficult to characterize the contact rate just by a constant or a proportion of the population size. In the present paper, latent age and a general nonlinear contact rate are proposed and incorporated into the classical SEIS epidemic model. Then, the existence of equilibrium and the global stability of disease free equilibrium are discussed, respectively. At last, an condition is obtained to guarantee the exponentially asymptotical stability of the endemic equilibrium and is verified by an example.

The remainder of this paper is organized as follows: Section 2 formulates an SEIS epidemic model; Section 3 studies the existence of each equilibrium and the global stability of the disease free equilibrium; Section 4 discusses the asymptotical stability of the endemic equilibrium; Section 5 makes the conclusions.

2 Model Formulation

In [17], Fan Meng, *et al.* separated the whole population into three classes: the susceptibles, the exposed and the infectives, formulated and qualitatively studied the following SEIS epidemic model

$$\begin{cases} \frac{dS(t)}{dt} = \Lambda - \mu S(t) - \beta S(t)I(t) + \gamma I(t), \\ \frac{dE(t)}{dt} = \lambda S(t)I(t) - (\mu + \epsilon)E(t), \\ \frac{dI(t)}{dt} = \epsilon E(t) - (\mu + \gamma + \alpha)I(t), \\ N(t) = S(t) + I(t) + E(t), \end{cases} \tag{1}$$

where $S(t)$, $E(t)$ and $I(t)$ denote the numbers of the susceptibles, the exposed and the infectives at time t, respectively, Λ is the input flow, μ is the natural mortality rate, β is the transmission rate, α is the death rate caused by disease, $1/\epsilon$ is the mean latent period, $1/\gamma$ is the mean infectious period, βSI is the bilinear incidence rate.

Obviously, the contact rate of system (1) is standard. In the following, neglecting the infection of the infectives and structuring $E(t)$, ϵ and β by the latent age, we focus our attention on model (1) again. For convenience, denotes $e(t,\tau)$ as the distribution function of $E(t)$ over latent age τ at time t, $\beta(\tau)$ and $\epsilon(\tau)$ as the distribution functions of β and ϵ over τ, respectively. Then, the exposed individuals $E(t)$ at time t equals to $\int_0^\infty e(t,\tau)d\tau$, the nonlinear incidence rate is characterized by $\frac{C(S(t),E(t),I(t))}{S(t)+E(t)+I(t)}S(t)\int_0^\infty \beta(\tau)e(t,\tau)d\tau$, where $C(S(t),E(t),I(t))$ is the contact rate (*i.e.*, the mean number contacts per individual per unit time), the infective individuals from the exposed is described by $\int_0^\infty \epsilon(\tau)e(t,\tau)d\tau$, and the continuous time dynamics of the exposed individuals is governed by a partial differential equation other than an ordinary equation (*i.e.*, the second equation in model (1) will be replaced by a partial differential equation of the distribution $e(t,\tau)$). Gathering up above discussion we derive an SEIS epidemic model with latent age and general nonlinear contact rate as follows

$$\begin{cases} \frac{dS(t)}{dt} = \Lambda - \mu S(t) - \frac{C(S(t),E(t),I(t))}{N(t)}S(t)\int_0^\infty \beta(\tau)e(t,\tau)d\tau + \gamma I(t), \\ \frac{\partial e(t,\tau)}{\partial t} + \frac{\partial e(t,\tau)}{\partial \tau} = -(\mu+\epsilon(\tau))e(t,\tau), \\ \frac{dI(t)}{dt} = \int_0^\infty \epsilon(\tau)e(t,\tau)d\tau - (\mu+\alpha+\gamma)I(t), \\ E(t) = \int_0^\infty e(t,\tau)d\tau, \\ N(t) = S(t)+E(t)+I(t), \\ e(t,0) = \frac{C(S(t),E(t),I(t))}{N(t)}S(t)\int_0^\infty \beta(\tau)e(t,\tau)d\tau, \\ e(0,\tau) = \eta(\tau), S(0) = S_0, I(0) = I_0, \end{cases} \tag{2}$$

where $\eta(\tau)$ is the initial distribution of the exposed individuals with latent age τ, S_0 and I_0 are the initial susceptibles and infectives, respectively.

Throughout the remainder of the paper, we adopt the following assumptions similar to Kim and Milner's [11]. The contact rate C is a nonnegative and partially differentiable function of S, E and I such that $\frac{\partial C}{\partial S}, \frac{\partial C}{\partial E}, \frac{\partial C}{\partial I} \in L^\infty([0,\infty)\times[0,\infty)\times[0,\infty))$; the nonnegative functions ϵ and β satisfy $\epsilon(\cdot) \in C^1[0,\infty)\bigcap L^\infty[0,\infty)$ with $\epsilon^{'}(\cdot) \in L^\infty[0,\infty)$, and $\beta(\cdot) \in C^2[0,\infty)\bigcap L^\infty[0,\infty)$ with $\beta^{'}(\cdot), \beta^{''}(\cdot) \in L^\infty[0,\infty)$; Λ, α, μ and γ are positive constants. Denote by $\|\cdot\|_1$ the norm of Banach space $L^1[0,\infty)$, by $\|\cdot\|_\infty$ the norm of Banach space $L^\infty[0,\infty)$, and by $L^1_+[0,\infty)$ the positive cone of nonnegative functions in $L^1[0,\infty)$, $\eta \in L^1_+[0,\infty)$.

3 Existence of Equilibrium

In epidemic dynamic, the existence and stability of equilibrium are important research topics because of the equilibrium standing for the possible ultima states of the special disease, and the asymptotic stability of an equilibrium revealing the capability of disease that tends to the ultima state. This section is mainly devoted to the existence of equilibrium and the asymptotic stability of disease free equilibrium of the model (2). Simultaneously, the reproductive number $\mathbb{R}_0$ (*namely*, the number of secondary cases produced in a completely susceptible population by a typical infected individual during its whole infected period) is obtained, which is usually intimately connected with the existence of equilibria.

For convenience, we introduce the following denotations:

$$B(t) = \frac{C(S(t), E(t), I(t))}{N(t)} S(t) \int_0^\infty \beta(\tau) e(t,\tau) d\tau,$$

$$\pi(\tau) = e^{-\int_0^\tau (\mu+\epsilon(s))ds}, \pi(\tau_1, \tau_2) = e^{-\int_{\tau_1}^{\tau_2} (\mu+\epsilon(s))ds}, m = \mu + \alpha + \gamma.$$

It is easy to verify that the equilibrium of system (2) has the following form

$$\begin{cases} S = \frac{\Lambda - B\left(1 - \frac{\gamma}{m}\int_0^\infty \pi(\tau)\epsilon(\tau)d\tau\right)}{\mu}, \\ E = B\int_0^\infty \pi(\tau)d\tau, \\ I = \frac{B}{m}\int_0^\infty \pi(\tau)\epsilon(\tau)d\tau, \\ B = \frac{C(S,E,I)}{N} SB \int_0^\infty \beta(\tau)\pi(\tau)d\tau. \end{cases} \tag{3}$$

Define a real function G on $[0, \infty)$

$$G(B) = \frac{C(S(B), E(B), I(B))}{S(B) + E(B) + I(B)} S(B) \int_0^\infty \beta(\tau)\pi(\tau)d\tau, \ B \geq 0, \tag{4}$$

where $S = S(B), E = E(B)$ and $I = I(B)$ are defined by (3). It is clear that G is a continuous real function of B.

Let

$$\mathbb{R}_0 = C\left(\frac{\Lambda}{\mu}, 0, 0\right) \int_0^\infty \beta(\tau)\pi(\tau)d\tau. \tag{5}$$

It is easy to see that system (2) always possesses the disease free equilibrium $(\frac{\Lambda}{\mu}, 0, 0)$. By using the fixed theorem, we have the following results

Theorem 1. *If* $\mathbb{R}_0 > 1$, *there exists at least an endemic equilibrium, while if* $\mathbb{R}_0 \leq 1$ *and the function* G *defined by* (4) *is strictly decrease, no endemic equilibrium exists.*

Remark 1. The restriction on G is not an extreme one. In fact, it is not hard to check that if $C(S, E, I) = C(N)$ and $C'(N) \geq 0$ then G is strictly decreasing, *i.e.*, the contact rate C is a function of the total population and increases with the population size.

In the remainder of this section, we study the asymptotic behavior of the disease free equilibrium. Using the method applied in [13], we obtain the global asymptotic stability of the disease free equilibrium under the condition of $\mathbb{R}_0 < 1$.

Theorem 2. *Assume that $C = C(N)$, $C'(N) \geq 0$ and $\mathbb{R}_0 < 1$, the disease free equilibrium of the model (2) is globally asymptotically stable.*

Further, by using the similar method adopted in [12], we obtain the following more general stability theorem.

Theorem 3. *Assume that $C(S, E, I) \leq C(\frac{\Lambda}{\mu}, 0, 0)$ for all nonnegative S, E and I. If $\mathbb{R}_0 < 1$, then the disease free equilibrium of the model (2) is globally asymptotically stable.*

The proofs for **Theorem 3.2-3.3** are trivial, omit them.

4 The Asymptotical Stability of the Endemic Equilibrium

In the present section, we discuss the asymptotical stability of the endemic equilibrium when it exists. Let $(S^*, e^*(\tau), I^*)$ be an endemic equilibrium of (2). Set $\overline{S}(t) = S(t) - S^*, \overline{e}(t,\tau) = e(t,\tau) - e^*(\tau)$ and $\overline{I}(t) = I(t) - I^*$. Then, model (2) is equivalently transformed into

$$\begin{cases} \frac{d\overline{S}(t)}{dt} = -\mu\overline{S}(t) + \gamma\overline{I}(t) - \overline{B}(t), \\ \frac{\partial\overline{e}(t,\tau)}{\partial t} + \frac{\partial\overline{e}(t,\tau)}{\partial\tau} = -(\mu + \epsilon(\tau))(\overline{e}(t,\tau) + e^*(\tau)), \\ \frac{d\overline{I}(t)}{dt} = \int_0^\infty \epsilon(\tau)\overline{e}(t,\tau)d\tau - m\overline{I}(t), \\ \overline{B}(t) = \overline{e}(t,0) = M(t)\int_0^\infty \beta(\tau)(\overline{e}(t,\tau) + e^*(\tau))d\tau - M^*\int_0^\infty \beta(\tau)e^*(\tau), \\ \overline{S}(0) = S_0 - S^*, \overline{\eta}(\tau) = \eta(\tau) - e^*(\tau), \overline{I}(0) = I(0) - I^*, \end{cases} \tag{6}$$

where $M(t) = \frac{C(S,E,I)S}{N}, M^* = \frac{C(S^*,E^*,I^*)S^*}{N^*}$.

Integrating the second equation of system (6) along the characteristic $t = \tau$, we get

$$\overline{e}(t,\tau) = \begin{cases} \overline{B}(t-\tau)\pi_(\tau), & t - \tau \geq 0; \\ \overline{\eta}(\tau - t)\pi(\tau - t, \tau), & \tau - t > 0. \end{cases} \tag{7}$$

Substituting (7) into (6) and integrating (6) with respect to τ from 0 to t lead to

$$\begin{cases} \overline{S}(t) = e^{-\mu t}\overline{S}(0) + \int_0^t e^{-\mu(t-\tau)}[\gamma\overline{I}(\tau) - \overline{B}(\tau)]d\tau, \\ \overline{E}(t) = \int_0^t [\overline{B}(t-\tau)\pi(\tau)d\tau + \int_t^\infty \overline{\eta}(\tau - t)\pi(\tau - t, \tau)d\tau, \\ \overline{I}(t) = e^{-mt}\overline{I}(0) + \int_0^t e^{-m(t-\tau)}\int_0^\tau \overline{B}(\tau - s)\pi(s)dsd\tau \\ \qquad + \int_0^t e^{-m(t-\tau)}\int_\tau^\infty \epsilon(s)\overline{\eta}(s-\tau)\pi(s-\tau, s)dsd\tau, \\ \overline{B}(t) = M^*\int_0^\infty \beta(\tau)\overline{e}(t,\tau)d\tau + \nabla M^* \cdot (\overline{S}(t), \overline{E}(t), \overline{I}(t))\int_0^\infty \beta(\tau)e^*(\tau)d\tau + \Psi(t) \end{cases} \tag{8}$$

with

$$\Psi(t) = [M(t) - M^* - \nabla M^* \cdot (\overline{S}(t), \overline{E}(t), \overline{I}(t))] \int_0^\infty \beta(\tau)(\overline{e}(t,\tau) + e^*(\tau))d\tau$$
$$+\nabla M^* \cdot (\overline{S}(t), \overline{E}(t), \overline{I}(t)) \int_0^\infty \beta(\tau)\overline{e}(t,\tau)d\tau.$$

Let $\overline{X}(t) = (\overline{S}(t), \overline{E}(t), \overline{I}(t), \overline{B}(t))^T$. It can be verified that system (8) is equivalent to the compact form

$$A\overline{X} + \int_0^t K(t-\tau)\overline{X}(\tau)d\tau = f(t). \tag{9}$$

It is a routine matter to verify that there exists a positive D such that

$$\|K(t)\|, \|K'(t)\|, \|K''(t)\| \leq De^{-\mu t}. \tag{10}$$

Obviously, the Laplace transform $\hat{K}(s)$ of $K(t)$ is analytic in the right half plane $\Re(s) > -\mu$. Moreover, it is easy to verify $\lim_{|s|\to+\infty} \hat{K}(s) = 0$ and then $\lim_{|s|\to+\infty} \det(A + \widehat{K}(s)) = 1$. Therefore, all of the roots of $\det(A + \widehat{K}(s))$ are isolated and lie in a certain ball centered at the origin.

Assume that all roots of $\det(A + \widehat{K}(s))$ have negative real parts, *that is*, there exists a μ^* and $0 < \mu^* < \mu$ such that no root of $\det(A + \widehat{K}(s))$ lies outside $\Re(s) < -\mu^*$. Let $L(s)$ denote the analytic reverse matrix of $A + \widehat{K}(s)$ in $\Re(s) \geq -\mu^*$. Since A is invertible and $\lim_{|s|\to+\infty} \hat{K}(s) = 0$, for sufficiently large $|s|$ and $\Re(s) > -\mu$ we have $L(s) = A^{-1}(I + A^{-1}\widehat{K}(s))^{-1} = A^{-1}\sum_{j=0}^\infty (A^{-1}\widehat{K}(s))^j$ and $\lim_{|s|\to\infty} L(s) = \lim_{|s|\to\infty} A^{-1}(I + A^{-1}\hat{K}(s))^{-1} = A^{-1}$. Moreover, by Taylor theorem it can be shown that $\hat{K}(s) = \frac{K(0)}{s} + \frac{K'(0)}{s^2} + o(s^{-2}), \quad$ for $|s| \to \infty$ in $\Re(s) > -\mu^*$. Therefore, we get a constant matrix J_0 such that $L(s) = A^{-1} + \frac{1}{s}J_0 + O(s^{-2}), \quad$ for $|s| \to \infty$ in $\Re(s) > \mu^*$. Which implies that $\widehat{J}(s) := L(s) - A^{-1}$ is the Laplace transform of $J(t)$, where

$$J(t) = \frac{1}{2\pi}e^{-\mu^* t}\int_{-\infty}^\infty e^{i\xi t}\hat{J}(-\mu^* + i\xi)d\xi, \quad t \geq 0. \tag{11}$$

It is easy to obtain from (10) that there exists a constant D_1 such that

$$\|J(t)\| \leq D_1 e^{-\mu^* t}, \quad t \geq 0.$$

To discuss the asymptotical stability of endemic equilibrium, we make the following assumption, which is a reduction of the assumption $H5$ of [12].

A. $|M - M^* - \nabla M^* \cdot (\overline{S}, \overline{E}, \overline{I})| = o(|\overline{S}| + |\overline{E}| + |\overline{I}|)$, as $|\overline{S}| + |\overline{E}| + |\overline{I}| \to 0$, *that is*, for $\epsilon_0 > 0$, there exists $\overline{\delta}(\epsilon_0) > 0$ such that $|\overline{S}| + |\overline{E} + |\overline{I}| < \overline{\delta}(\epsilon_0)$, $|M - M^* - \nabla M^* \cdot (\overline{S}, \overline{E}, \overline{I})| < \epsilon_0(|\overline{S}| + |\overline{E}| + |\overline{I}|)$.

Remark 2. It should be pointed out that this assumption is obviously satisfied if $C(S,E,I) = \beta_1 N$, where β_1 is nonnegative and N is the population size.

Theorem 4. *Under the assumption* **A**. *If all roots of* $det(A+\hat{K}(s))$ *have negative real parts, then there exist positive numbers* a, b *and* δ *such that for initial value* $S_0, \eta(\tau)$, I_0 *with* $|S_0|+|I_0|+\|\eta\|_1<\delta$, *the solutions for* (2) *satisfy* $|S(t)-S^*|+\|e(t,\cdot)-e^*(\cdot)\|_1+|I(t)-I^*|+|e(t,0)-e^*(0)|<ae^{-bt}, t\geq 0$.

Proof. Denote by $\widehat{\overline{X}}(s)$ the Laplace transform of $\overline{X}(t)$ and by $\widehat{f}(s)$ the Laplace transform of $f(t)$. Then, by using the convolution formulas of Laplace transform, (9) can be changed into $A\widehat{\overline{X}}(s)+\widehat{K}(s)\widehat{\overline{X}}(s)=\widehat{f}(s)$. Hence, we have $\overline{X}(t)=A^{-1}f(t)+\int_0^\infty J(t-\tau)f(\tau)d\tau, \quad t\geq 0$, where $\mathfrak{L}^{-1}$ represents the inverse Laplace transform. To obtain a bound of $\overline{X}(t)$, we consider the following system

$$\begin{cases} \frac{d\underline{S}(t)}{dt}=-\mu\underline{S}(t)+\gamma\underline{I}(t)-\underline{B}(t),\\ \frac{\partial\underline{e}(t,\tau)}{\partial t}+\frac{\partial\underline{e}(t,\tau)}{\partial\tau}=-(\mu+\epsilon(\tau))\underline{e}(t,\tau),\\ \frac{d\underline{I}(t)}{dt}=\int_0^\infty\epsilon(\tau)\underline{e}(t,\tau)d\tau-m\overline{I}(t),\\ \underline{B}(t)=\underline{e}(t,0)=M^*\int_0^\infty\beta(\tau)\underline{e}(t,\tau)d\tau+\nabla M^*\cdot(\underline{S}(t),\underline{E}(t),\underline{I}(t)),\\ \underline{S}(0)=\overline{S}(0),\underline{I}(0)=\overline{I}(0),\underline{\eta}(\tau)=\overline{\eta}(\tau). \end{cases} \tag{12}$$

Clearly,

$$\underline{e}(t,\tau)=\begin{cases}\underline{B}(t-\tau)\pi(\tau), & t-\tau\geq 0;\\ \overline{\eta}(\tau-t)\pi(\tau-t,\tau), & \tau-t>0,\end{cases}$$

where $\underline{B}(t)=\underline{e}(t,0)$. Let $\underline{Y}(t)=(\underline{S}(t),\underline{E}(t),\underline{I}(t),\underline{B}(t))^T$. Then, system (12) can be written as

$$A\underline{Y}(t)+\int_0^t K(t-\tau)\underline{Y}(\tau)d\tau=\underline{l}(t), \tag{13}$$

where

$$\underline{l}(t)=\begin{pmatrix} e^{-\mu t}\overline{S}(0)\\ \int_t^\infty\overline{\eta}(\tau-t)\pi(\tau-t,\tau)d\tau\\ e^{-mt}\overline{I}(0)+\int_0^t e^{-m(t-\tau)}\int_\tau^\infty\epsilon(\rho)\overline{\eta}(\rho-\tau)\pi(\rho-\tau,\rho)d\rho d\tau\\ M^*\int_t^\infty\beta(\tau)\overline{\eta}(\tau-t)\pi(\tau-t,\tau)d\tau\end{pmatrix}. \tag{14}$$

and A and K are defined by (9). Similar to the previous discussion, we have

$$\underline{Y}(t)=A^{-1}\underline{l}(t)+\int_0^t J(t-\tau)\underline{l}(\tau)d\tau. \tag{15}$$

Let us endow 4-dimensional real space $\mathbb{R}^4$ with the norm $\|x\|=\sum_{j=1}^{4}|x_j|$. Then, it follows from (14) that

$$\|\underline{l}\|\leq\left(1+M^*\beta_\infty+\frac{\epsilon_\infty}{b-\mu}\right)\overline{N}(0)e^{-\mu t}:=D_2\overline{N}(0)e^{-\mu t}, \tag{16}$$

where $\overline{N}(0) = |\overline{S}(0)| + |\overline{I}(0)| + \|\overline{\eta}\|_1$, $D_2 = 1 + M^*\beta_\infty + \frac{\epsilon_\infty}{b-\mu}$, β_∞, ϵ_∞ are the norms of β, γ in $L^\infty[0,\infty)$.

Considering (15), (16) and the second equation of (12), we get the following estimations of $\|\underline{Y}\|$ and $\|\underline{e}(t,\cdot)\|$

$$\|\underline{Y}\| \le D_2\left[|A^{-1}| + \frac{D_1}{\mu-\mu^*}\right]\overline{N}(0)e^{-\mu^* t} := D_3\overline{N}(0)e^{-\mu^* t}, \tag{17}$$

$$\|\underline{e}(t,\cdot)\|_1 \le 1 + \frac{D_3}{\mu-\mu^*})\overline{N}(0)e^{-\mu^* t} := D_4\overline{N}(0)e^{-\mu^* t}, \tag{18}$$

where $D_4 = 1 + \frac{D_3}{\mu-\mu^*}$. Let $W(t) = |\overline{S}(t)| + |\overline{E}(t)| + |\overline{I}(t)| + \|\overline{e}(t,\cdot)\|_1$. Without loss of generality, for some ϵ_0, choosing $\overline{\delta}(\epsilon_0) < \epsilon_0$, under the assumption A, we get

$$|\Psi(t)| \le \epsilon_0(D_5 + D_6)W(t) := \epsilon_0 D_7 W(t), \tag{19}$$

where $D_5 = \beta_\infty(\epsilon_0 + \max\{|M_j^*|, j = 1,2,3\})$, $D_6 = \int_0^\infty \beta(\tau)e^*(\tau)d\tau$. By (19) , we obtain

$$\|f(t) - \underline{l}(t)\| \le \epsilon_0 D_7 W(t). \tag{20}$$

Note that

$$\overline{X}(t) = \underline{Y}(t) + A^{-1}(f(t) - \underline{l}(t)) + \textstyle\int_0^t J(t-\tau)(f(\tau) - \underline{l}(\tau))d\tau,$$

we get

$$\|\overline{X}\| \le D_3\overline{N}(0)e^{-\mu^* t} + \epsilon_0 D_9\int_0^t e^{-\mu^*(t-\tau)}W(\tau)d\tau + \epsilon_0 D_{10}W(t) \tag{21}$$

and

$$\|\overline{X}(t) - \underline{Y}(t)\| \le \epsilon_0 D_9\int_0^t e^{-\mu^*(t-\tau)}W(\tau)d\tau + \epsilon_0 D_{10}W(t), \tag{22}$$

where $D_9 = D_1D_7$, $D_{10} = |A^{-1}|D_7$. Obviously, we have

$$\int_0^t \|\overline{X}(\tau) - \underline{Y}(\tau)\|e^{-\mu(t-\tau)}d\tau \le \epsilon_0\left(\frac{D_{10}}{\mu-\mu^*} + D_9\right)\int_0^t e^{-\mu^*(t-\tau)}W(\tau)d\tau. \tag{23}$$

By using (18), (22), (23), we have

$$\|\overline{e}(t,\cdot)\|_1 \le D_4\overline{N}(0)e^{-\mu^* t} + \epsilon_0 D_{13}\int_0^t e^{-\mu^*(t-\tau)}W(\tau)d\tau, \tag{24}$$

where $D_{13} = \frac{D_{10}}{\mu-\mu^*} + D_9$. Then, we have

$$W(t) \le \epsilon_0(D_9 + D_{13})\int_0^t e^{-\mu^*(t-\tau)}W(\tau)d\tau + (D_3 + D_4)\overline{N}(0)e^{-\mu^* t} + \epsilon_0 D_{10}W(t).$$

Let $\epsilon_0 < \frac{1}{D_{10}}$. It follows from Gronwall's lemma that

$$W(t) \le \frac{D_3 + D_4}{1 - \epsilon_0 D_{10}} \overline{N}(0) e^{-\mu^* t + \epsilon_0 \frac{D_9 + D_{13}}{1 - \epsilon_0 D_{10}} t}. \tag{25}$$

Substituting (17) and (25) into (21) yields

$$\|\overline{X}\| \le \epsilon_0 D_9 \int_0^t e^{-\mu^*(t-\tau)} \frac{D_3 + D_4}{1 - \epsilon_0 D_{10}} \overline{N}(0) e^{-(\mu^* - \frac{\epsilon_0 (D_9 + D_{13})}{1 - \epsilon_0 D_{10}})} d\tau$$
$$+ \epsilon_0 D_{10} \frac{D_3 + D_4}{1 - \epsilon_0 D_{10}} \overline{N}(0) e^{-(\mu^* - \frac{\epsilon_0 (D_9 + D_{13})}{1 - \epsilon_0 D_{10}})t} + D_3 \overline{N}(0) e^{-\mu^* t}.$$

Clearly, there exists positive constant ξ such that

$$\|\overline{X}(t)\| \le \xi \overline{N}(0) e^{-(\mu^* - \frac{\epsilon_0 (D_9 + D_{13})}{1 - \epsilon_0 D_{10}})t}. \tag{26}$$

Then, it is easy to obtain that there exist positive constants a, b, δ, such that $\|\overline{X}(t)\| < ae^{-bt}, t \ge 0$, if $|S_0| + \|\eta\|_1 + |I_0| < \delta$. This completes the proof. $\square$

Example 1. Let $\epsilon(\tau) \equiv \epsilon$, $\beta(\tau) = e^{-\beta\tau}$ and $C(S, E, I) = \beta' N$, where ϵ, β and β' are positive numbers. Correspondingly, we have $M = \beta' S$, $R_0 = \frac{\beta' \Lambda}{\mu(\mu+\epsilon+\beta)}$, $S^* = \frac{\mu+\epsilon+\beta}{\beta'}$, $B^* = \frac{m\mu(\mu+\epsilon)(\mu+\epsilon+\beta)(R_0-1)}{\beta'(m(\mu+\epsilon)-\gamma\epsilon)}$, $I^* = \frac{\epsilon B^*}{m(\mu+\epsilon)}$, $E^* = \frac{B^*}{\mu+\epsilon}$. Then, we further have $\det(A + \hat{K}(s)) = \frac{s^4 + a_1 s^3 + a_2 s^2 + a_3 s + a_4}{(s+m)(s+\mu)(s+\mu_1)(s+\mu_2)}$, where $a_1 = k + \epsilon + \alpha + \gamma + 3\mu$, $a_2 = 2k\epsilon + k\alpha + 2\mu\alpha + k\gamma + \epsilon\gamma + \epsilon\alpha + 3k\mu + k\beta + 2\epsilon\mu + 3\mu^2 + 2\mu\gamma$, $a_3 = 2k\mu\beta + 2k\epsilon\alpha + k\epsilon^2 + 3k\mu^2 + \mu^2\gamma + k\gamma\beta + 2k\mu\gamma + k\epsilon\beta + k\epsilon\gamma + 4k\epsilon\mu + \mu\alpha\epsilon + \mu\gamma\epsilon + 2k\mu\alpha + \mu^3 + k\alpha\beta + \mu^2\alpha + \mu^2\epsilon$, $a_4 = k(\beta + \mu + \epsilon)(\epsilon\mu + \epsilon\alpha + \mu^2 + \mu\alpha + \mu\gamma)$.

For convenience, we introduce the following denotations

$$\Omega_1 = a_1, \Omega_2 = \begin{pmatrix} a_1 & a_3 \\ 1 & a_2 \end{pmatrix}, \Omega_3 = \begin{pmatrix} a_1 & a_3 & 0 \\ 1 & a_2 & a_4 \\ 0 & a_1 & a_3 \end{pmatrix} \text{ and } \Omega_4 = \begin{pmatrix} a_1 & a_3 & 0 & 0 \\ 1 & a_2 & a_4 & 0 \\ 0 & a_1 & a_3 & 0 \\ 0 & 1 & a_2 & a_4 \end{pmatrix}.$$

It is easy to verify that $a_j > 0$ and $\det(\Omega_j) > 0, j = 1, 2, 3, 4$ if $R_0 > 1$. By Routh-Hurwitz Criterion, we get that all of roots of $s^4 + a_1 s^3 + a_2 s^2 + a_3 s + a_4$ have negative real parts. Then, we arrive at that all of roots of $\det(A + \hat{K}(s))$ have negative real parts.

5 Conclusion

In this paper, by introducing latent age and generally nonlinear contact rate, an SEIS epidemic model is formulated. It is mentionable that the incidence rate dependents on the latent age in our model, which makes the model be more suitable for the disease with long latent period such as AIDS, *et al.* Then, by fixed theorem and Lyapunov method, the existence of equilibrium and the globally asymptotic stability of the disease free equilibrium are discussed, respectively. Further then, as a main result of the paper a general condition is obtained to characterize the exponentially asymptotical stability of the endemic equilibrium. At last, this condition is verified by a special example.

References

1. Castillo-Chavez, C., Hethcote, H.W., Andreasen, V., Levin, S.A., Liu, W.M.: Epidemiological models with age structure, proportionate mixing, and cross-immunity. J. Math. Biol. 27, 233–258 (1981)
2. Hethcote, H.W., Yorke, J.A.: Gonorrhea transmission dynamics and control. LNCS (LNBI). Springer, Berlin (1984)
3. Francis, D.P., Feorino, P.M., Broderson, J.R., Mccluer, H.M., Getchell, J.P., Mcgrath, C.R., Swenson, B., Mcdugal, J.S., Palmer, E.L., Harrison, A.K., et al.: Infection of chimpanzees with lymphadenopathy-associated virus. Lancet. 2, 1276–1277 (1984)
4. Lange, J.M., Paul, D.A., Huisman, H.G., De Wolf, F., Van den Berg, H., et al.: Persistent HIV antigenaemia and decline of HIV core antibodies associated with transition to AIDS. British Medical J. 293, 1459–1462 (1986)
5. Kermack, W.O., Mckendrick, A.G.: A contribution to the mathematical theory of epidemics. Proc. Roy. Soc. Lond Ser. A 115, 700–721 (1927)
6. Mckendrick, A.G.: Applications of mathematics to medical problems. Proc. Edinburgh Math. Soc. 44, 98–130 (1926)
7. Kermack, W.O., Mckendrick, A.G.: Contributions to the mathematical theory of epidemics.II. The problem of endemictity. Proc. Roy. Soc. Lond Ser. A 138, 55–83 (1932)
8. Kermack, W.O., Mckendrick, A.G.: Contributions to the mathematical theory of epidemics.III. Further studies of the provlem of endemicity. Proc. Roy. Soc. Lond Ser. A 141, 94–122 (1932)
9. Hoppensteadt, F.: An age dependent epidemic model. J. Franklin Inst. 297, 325–333 (1974)
10. Hoppensteadt, F.: Mathematical theories of populations: demographics, genetics and epidemics. In: Regional Conference Deries in Applied Mathematics. Society for industrial and applied mathematics, Philadelphia (1975)
11. Kim, M.Y., Milner, F.A.: A mathematical model of epidmimcs with screening and variable infectivity. Mathl. Comput. Modelling 21, 29–42 (1995)
12. Kim, M.Y.: Existence of steady state solutions to an epidemic model with screening and their asymptotic stability. Appl. Math. Comput. 74, 37–58 (1974)
13. Thieme, H.R., Castillo-Chavez, C.: How infection-age-dependent infectivity affect the dynamics of HIV/AIDS? Siam J Appl. Math. 53, 1447–1479 (1993)
14. Kribs-Zaleta, C.M., Martcheva, M.: Vaccination strategies and backward bifurcation in an age-since-infection structured model. Math. Biosci. 177/178, 317–332 (2002)
15. Inaba, H., Sekine, H.: A mathematical model for Chagas disease wtih infection-age-dependent infectivity. Math. Biosci. 190, 39–69 (2004)
16. Li, J., Zhou, Y.C., Ma, Z.Z., Hyman, M.: Epedemiological models for mutating pathogens. Siam J. Appl. Math. 65, 1–23 (2004)
17. Fan, M., Li, M.Y., Wang, K.: Global stability of an SEIS epidemic model with recruitment and a varying total population size. Mathematical Biosciences 170, 199–208 (2001)

Stochastic Exponential Stability of Cohen-Grossberg Neural Networks with Markovian Jumping Parameters and Mixed Delays

Xiaohui Xu*, Jiye Zhang, and Weihua Zhang

Traction Power State Key Laboratory, Southwest Jiaotong University,
Chengdu 610031, China
xhxu@163.com

Abstract. In this paper by applying vector Lyapunov function method and M matrix theory which are different from all the existing study methods (LMI technique), some sufficient conditions ensuring stochastic exponential stability of the equilibrium point of a class of Cohen-Grossberg neural networks with Markovian jumping parameters and mixed delays are derived.

Keywords: Cohen-Grossberg neural networks, stochastic exponential stability, mixed delays, Markovian jumping, vector Lyapunov function.

1 Introduction

In recent years, considering that time delays which may lead to instability of networks unavoidably exist in neural network system, lots of papers about stability of the equilibrium point of neural network systems with time-delays have been emerged, see [1-6] and the references therein. The neural networks systems in papers [1-6] were assumed that the continuous variables propagate from one processing unit to the next. However, a phenomenon of information latching frequently happens in neural network systems. Therefore, neural network systems with Marvokian jumps attracted increasing attention over the last decade, these systems have the advantage of modeling dynamic systems subject to abrupt variation in their structures, such as component failures or repairs, sudden environmental disturbance, changing subsystem interconnections, and operating in different points of a nonlinear plant [7]. Recently, some pioneering works on stability analysis for neural network systems with Marvokian jumping parameters with time delays have been presented; see [7-12]. However, all the authors in [7-12] adopted LMI (Linear Matrix Inequality) technique to analyze the stochastic stability of neural network systems with Marvokian jumping parameters.

Motivated by the above analysis, in this paper it is the first attempt to use another study method named vector Lyapunov function method to discuss the stochastic exponential stability of the equilibrium point of a class of Cohen-Grossberg neural networks with Marvokian jumping parameters and mixed delays.

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

2 Model Description

For the sake of convenience some notions are introduced as follows.

For matrix $A=(a_{ij})_{n\times n}$, $|A|$ denotes the absolute-value matrix given by $|A|=(|a_{ij}|)_{n\times n}$; we define $A^*=(a_{ij}^*)_{n\times n}$, where $a_{ii}^*=a_{ii}$ as $a_{ii}\ge 0$, and $a_{ii}^*=0$ as $a_{ii}<0$, $a_{ij}^*=|a_{ij}|$ $(i\ne j)$.

As we known, Cohen-Grossberg neural networks (CGNN) proposed in 1983 [13] is a kind of generalized neural networks because it includes the well-known Hopfield neural networks, Cellular neural networks and BAM neural networks and so on, therefore in this paper we will mainly study the stochastic exponential stability of CGNN system with mixed delays and Markovian jumping parameters which can be described by the following equations:

$$\dot{u}_i(t)=-\alpha_i(u_i(t))\{d_i(r(t))u_i(t)-\sum_{j=1}^{n}[a_{ij}(r(t))g_j(u_j(t))+b_{ij}(r(t))g_j(u_j(t-\tau_{ij}(t)))$$
$$+c_{ij}(r(t))\int_{-\infty}^{t}\sigma_{ij}(t-s)g_j(u_j(s))ds]+J_i\},\ t\ge 0,\ i=1,2,...,n, \tag{1}$$

where $n\ge 2$ denotes the number of neurons, $u_i(t)$ represents the state variable of i-th neuron, $i=1,2,...,n$. $d_i(r(t))$ denotes the charging time constant or passive decay rate of the n-th neuron, $g(u)=(g_1(u_1),g_2(u_2),...,g_n(u_n))^T$ corresponds to the activation function of neurons. $\tau_{ij}(t)\ge 0$, $i,j=1,2,...,n$, is time-varying delays of the neural networks, and $\tau=\sup_{1\le i,j\le n,t\in R}\{\tau_{ij}(t)\}$. $\sigma_{ij}:[0,\infty)\to[0,\infty)$ is piecewise continuous on $[0,\infty)$ and satisfies

$$\int_0^{\infty}e^{\beta s}\sigma_{ij}(s)ds=\rho_{ij}(\beta),\ i,j=1,2,...,n, \tag{2}$$

where $\rho_{ij}(\beta)$ is continuous function in $[0,\ \delta]$, here $\delta>0$, and $\rho_{ij}(0)=1$. J_i denotes the external input on the i-th neuron, let $J=(J_1,J_2,...,J_n)^T$. $A(r(t))=[a_{ij}(r(t))]_{n\times n}$, $B(r(t))=[b_{ij}(r(t))]_{n\times n}$ and $C(r(t))=[c_{ij}(r(t))]_{n\times n}$ represent the weight matrices of the neurons, where $A(r(t))$, $B(r(t))$ and $C(r(t))$ are given matrices for each value of $r(t)$ in a finite set $S=\{1,2,...,N\}$ with the initial mode and the initial value are $r(0)=r_0$. Let $\{r(t),t>0\}$ that determines the mode of the system at each time *t* be described by the probability transitions with transition probability parameters given by following equation:

$$\mathrm{P}\{r(t+\Delta t)=l\mid r(t)=k\}=\begin{cases}\pi_{kl}\Delta t+\mathrm{O}(\Delta t), & k\ne l\\ 1+\pi_{kl}\Delta t+\mathrm{O}(\Delta t), & k=l\end{cases}, \tag{3}$$

where $\Delta t>0$, $\pi_{kl}>0$, $\pi_{kk}=-\sum_{l=1,l\ne k}^{N}\pi_{kl}$ for $k,l\in S$, $k\ne l$, and $\lim_{\Delta t\to 0}\mathrm{O}(\Delta t)/\Delta t=0$.

The initial conditions of (1) are of the forms $u_i(s)=\varphi_i(s)$, $s\le 0$, where $\varphi_i(s)$ is continuous and bounded on $(-\infty,0]$.

Let $u^*=(u_1^*,u_2^*,\cdots,u_n^*)^T$ be the equilibrium point of system (1).

Definition 1. The equilibrium point u^* of system (1) is said to be stochastically exponentially stable if for every system mode there exist constants $M(k) > 0$ ($k \in S$) and $\lambda > 0$ such that for all $J \in R^n$, the following inequalities hold:

$$\mathrm{E}\{|u_i(t) - u_i^*, r(t) = k|\} \le M(k) E\{|\varphi(s) - u^*, r_0|\} \mathrm{e}^{-\lambda t},$$

where $|\varphi(s) - u^*, r_0| = \max_{1 \le i \le n} \sup_{s \in (-\infty, 0]} |\varphi_i(s) - u_i^*, r_0|$, $i = 1,2,...,n$, $t \ge 0$.

Definition 2. [4] 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.

Next, we will give some assumptions for system (1).

Assumption A1. 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 , the following inequalities hold:

$$|g_i(u_i) - g_i(v_i)| \le L_i |u_i - v_i| .$$

Assumption A2. For each $i \in \{1,2,...,n\}$, $\alpha_i : R \to R$ is a continuous function and $0 < \underline{\theta}_i \le \alpha_i \le \overline{\theta}_i$, where $\underline{\theta}_i$, $\overline{\theta}_i$ are constants.

3 Main Results

In this section, we will establish a family of sufficient conditions for any $k \in S$ ensuring stochastic exponential stability of the equilibrium point of (1).

For the purpose of simplicity, we introduce coordinate translation for system (1), let $x_i = u_i - u_i^*$, $i = 1,2,...,n$, the system (1) can be transformed into the following equations:

$$\dot{x}_i(t) = -\beta_i(x_i(t))\{d_i(r(t))x_i(t) - \sum_{j=1}^{n}[a_{ij}(r(t))f_j(x_j(t)) + b_{ij}(r(t))f_j(u_j(t - \tau_{ij}(t)))$$
$$+ c_{ij}(r(t))\int_{-\infty}^{t} \sigma_{ij}(t-s)f_j(u_j(s))ds]\} \quad t \ge 0 ,\ i = 1,2,...,n , \tag{4}$$

where

$$\beta_i(x_i(t)) = \alpha_i(x_i(t) + u_i^*) ;$$
$$f_j(x_j(t)) = g_j(x_j(t) + u_j^*) - g_j(u_j^*) .$$

The initial condition of (4) is $\phi_i(s) = \varphi_i(s) - u_i^*$, $-\infty < s \le 0$, $i = 1,2,...,n$. It is obvious that $\beta_i(x_i(t))$ is continuous and we have $0 < \underline{\theta}_i \le \beta_i \le \overline{\theta}_i$.

Next, we will give some sufficient conditions for (4).

Theorem 1. Suppose that Assumption A1- A2 are satisfied, then the zero solution of system (4) is stochastically exponentially stable if there exist a sequence of positive scalars ω_i^k, $i=1,2,...,n$, $k\in S$, such that every matrix $Q(k)$ ($k\in S$) is a M-matrix, where

$$q_{ii}^k = -\sum_{l=1}^{N}\pi_{kl}\omega_i^l(\omega_i^k)^{-1} + \underline{\theta}_i d_i^k ;$$

$$q_{ij}^k = -\overline{\theta}_i\sum_{j=1}^{n}\omega_i^k(\omega_j^k)^{-1}\gamma_j[(a_{ij}^k)^* + |b_{ij}^k| + |c_{ij}^k|],\ i,j=1,2,...,n .$$

Proof. Firstly, when the mode of system (4) is decided, the existence and uniqueness of the equilibrium point of system (4) can be obtained directly by using the methods in literature [3, 4].

Next, we will discuss the stochastic exponential stability of system (4).

Because $Q(k)=(q_{ij}^k)_{n\times n}$ is a M matrix for every $k\in S$, we know from the properties of M-matrix [4] that there exists positive vector $\xi(k)=(\xi_1^k,\xi_2^k,...,\xi_n^k)^T$ such that

$$[\sum_{l=1}^{N}\pi_{kl}\omega_i^l(\omega_i^k)^{-1} - \underline{\theta}_i d_i^k]\xi_i^k + \overline{\theta}_i\sum_{j=1}^{n}\omega_i^k(\omega_j^k)^{-1}\gamma_j\xi_j^k[(a_{ij}^k)^* + |b_{ij}^k| + |c_{ij}^k|] < 0 . \tag{5}$$

Let

$$F_i(\mu,k) = [\sum_{l=1}^{N}\pi_{kl}\omega_i^l(\omega_i^k)^{-1} - \underline{\theta}_i d_i^k + \mu]\xi_i^k$$
$$+\overline{\theta}_i\sum_{j=1}^{n}\omega_i^k(\omega_j^k)^{-1}\gamma_j\xi_j^k[(a_{ij}^k)^* + e^{\mu\tau}|b_{ij}^k| + \rho_{ij}(\mu)|c_{ij}^k|],\ i=1,2,...,n ,$$

it follows from (5) that

$$F_i(0,k) = [\sum_{l=1}^{N}\pi_{kl}\omega_i^l(\omega_i^k)^{-1} - \underline{\theta}_i d_i^k]\xi_i^k + \overline{\theta}_i\sum_{j=1}^{n}\omega_i^k(\omega_j^k)^{-1}\gamma_j\xi_j^k[(a_{ij}^k)^* + |b_{ij}^k| + |c_{ij}^k|] < 0 ,$$

so there exists $\lambda>0$ such that

$$F_i(\lambda,k) = [\sum_{l=1}^{N}\pi_{kl}\omega_i^l(\omega_i^k)^{-1} - \underline{\theta}_i d_i^k + \lambda]\xi_i^k$$
$$+\overline{\theta}_i\sum_{j=1}^{n}\omega_i^k(\omega_j^k)^{-1}\gamma_j\xi_j^k[(a_{ij}^k)^* + e^{\lambda\tau}|b_{ij}^k| + \rho_{ij}(\lambda)|c_{ij}^k|],\ i=1,2,...,n . \tag{6}$$

Let $V_i(t,k)=e^{\lambda t}\omega_i^k|x_i|$, $i=1,2,...,n$, calculating the upper right derivation $L_{(4)}V_i(t,k)$ along the solution of (4), we get

$$L_{(4)}V_i(t,k) = \sum_{l=1}^{N}\pi_{kl}V_i(t,l) + D^+V_i(t,k)$$
$$= \sum_{l=1}^{N}\pi_{kl}V_i(t,l) + \lambda V_i(t,k) + e^{\lambda t}\omega_i^k\,\mathrm{sgn}\{x_i\}\{-\beta_i(x_i)d_i^k x_i$$
$$+\beta_i(x_i)\sum_{j=1}^{n}[a_{ij}^k f_j(x_j(t)) + b_{ij}^k f_j(x_j(t-\tau_{ij}(t))) + c_{ij}^k\int_{-\infty}^{t}\sigma_{ij}(t-s)f_j(x_j(s))ds]\} .$$

From Assumption A1-A2, we have

$$L_{(4)}V_i(t,k) \le \sum_{l=1}^{N} \pi_{kl} V_i(t,l) + \lambda V_i(t,k) + \mathrm{e}^{\lambda t}\, \omega_i^k \{-\underline{\theta}_i d_i^k \,|\, x_i \,| - \overline{\theta}_i(x_i) \sum_{j=1}^{n} \gamma_j [(a_{ij}^k)^* \,|\, x_j(t)\,|$$

$$+ |\, b_{ij}^k \,| \cdot |\, x_j(t-\tau_{ij}(t)) \,| + |\, c_{ij}^k \,| \int_{-\infty}^{t} \sigma_{ij}(t-s) \,|\, x_j(s) \,|\, ds]\}$$

$$\le \sum_{l=1}^{N} \pi_{kl} \omega_i^l (\omega_i^k)^{-1} V_i(t,k) + \lambda V_i(t,k) - \underline{\theta}_i d_i^k V_i(t,k) + \overline{\theta}_i \sum_{j=1}^{n} \gamma_j \omega_i^k (\omega_j^k)^{-1} [(a_{ij}^k)^* V_j(t)$$

$$+ |\, b_{ij}^k \,| V_j(t-\tau_{ij}(t)) + |\, c_{ij}^k \,| \int_{-\infty}^{t} \sigma_{ij}(t-s) e^{\lambda(t-s)} V_j(s,k) ds] . \tag{7}$$

Computing the expectation on both sides of (7), we have

$$\mathrm{E}\{L_{(4)}V_i(t,k)\} \le \{\sum_{l=1}^{N} \pi_{kl} \omega_i^l (\omega_i^k)^{-1} - \underline{\theta}_i d_i^k + \lambda\} \mathrm{E}[V_i(t,k)] + \overline{\theta}_i \sum_{j=1}^{n} \gamma_j \omega_i^k (\omega_j^k)^{-1} \{(a_{ij}^k)^* \mathrm{E}[V_j(t)]$$

$$+ |\, b_{ij}^k \,| \mathrm{E}[V_j(t-\tau_{ij}(t))] + |\, c_{ij}^k \,| \int_{-\infty}^{t} \sigma_{ij}(t-s) e^{\lambda(t-s)} \mathrm{E}[V_j(s,k)] ds\} , \tag{8}$$

Defining the curve

$$\zeta(k) = \{ y(\chi,k) : y_i(\chi,k) = \xi_i^k \chi, \chi > 0\}$$

and the sets

$$\Lambda(k) = \{ z : 0 \le z \le y, y \in \zeta(k)\} ,$$

here $k \in S$. Let $\xi_{\min}^k = \min_{1\le i\le n}\{\xi_i^k\}$, $\xi_{\max}^k = \max_{1\le i\le n}\{\xi_i^k\}$, taking $\chi_0(k) = \delta^k \mathrm{E}\{|\phi|\} / \xi_{\min}^k$, here $\delta^k > 1$ is a constant.

Obviously, we have

$$\{\mathrm{E}\{V(s,k)\} : \mathrm{E}\{V\} = \mathrm{e}^{\lambda s}\, \mathrm{E}\{|\phi|\}, -\tau \le s \le 0\} \subset \Lambda(y(\chi_0(k),k)) ,$$

i.e.

$$\mathrm{E}\{V_i(s,k)\} < \xi_i(k) \chi_0(k) ,\ -\tau \le s \le 0 ,\ i = 1,2,...,n .$$

We claim that $\mathrm{E}\{V_i(t,k)\} < \xi_i^k \chi_0(k)$, $t \ge 0$, $i = 1,2,...,n$. If it is not true, then there exists some i and $t' > 0$, such that $\mathrm{E}\{V_i(t',k)\} = \xi_i^k \chi_0(k)$, $\mathrm{E}\{L_{(4)}V_i(t',k)\} \ge 0$ and $\mathrm{E}\{V_j(t,k)\} \le \xi_j^k \chi_0(k)$, $-\infty < t \le t'$, $j = 1,2,...,n$. However, from (6) and (8) we get

$$\mathrm{E}\{L_{(4)}V_i(t',k)\} \le [\sum_{l=1}^{N} \pi_{kl} \omega_i^l (\omega_i^k)^{-1} - \underline{\theta}_i d_i^k + \lambda] \xi_i^k \chi_0(k)$$

$$+ \overline{\theta}_i \sum_{j=1}^{n} \omega_i^k (\omega_j^k)^{-1} \gamma_j \xi_j^k \chi_0(k) [(a_{ij}^k)^* + \mathrm{e}^{\lambda\tau} |\, b_{ij}^k \,| + \rho_{ij}(\lambda) |\, c_{ij}^k \,|] < 0 ,$$

This is a contradiction, therefore $\mathrm{E}\{V_i(t,k)\} < \xi_i^k \chi_0(k)$, *i.e.*

$$\mathrm{E}\{|\, x_i(t) \,|, k\} < \{(\omega_i^k)^{-1} \xi_i^k \delta(k) \mathrm{E}\{|\phi|\} / \xi_{\min}^k\} \mathrm{e}^{-\lambda t} = M(k) \mathrm{E}\{|\phi|\} \mathrm{e}^{-\lambda t} , \tag{9}$$

here $M(k) = (\omega_i^k)^{-1} \xi_i^k \delta(k) / \xi_{\min}^k$.

From (9) and Definition 1, the zero solution of the system (4) is stochastically exponentially stable, namely, the equilibrium point of (1) is stochastically exponentially stable. The proof is completed.□

4 Example

For the sake of convenience, consider the following 2-dimension neural networks system with three modes and mixed delays.

$$\begin{cases} \dot{u}_1(t,x) = [\sin(u_1(t))+1.5]\{-d_1(r(t))u_1(t)+a_{11}(r(t))\tanh(u_1(t)) \\ \qquad +a_{12}(r(t))\tanh(u_2(t))+b_{11}(r(t))\tanh(u_1(t-\tau_{ij}(t))) \\ \qquad +b_{12}(r(t))\tanh(u_2(t-\tau_{ij}(t)))\} \\ \qquad +c_{11}(r(t))\int_{-\infty}^{t} e^{t-s}\tanh(u_1(s))ds+c_{12}(r(t))\int_{-\infty}^{t} e^{t-s}\tanh(u_2(s))ds \\ \dot{u}_2(t,x) = [\cos(u_1(t))+2]\{-d_2(r(t))u_1(t)+a_{21}(r(t))\tanh(u_1(t)) \\ \qquad +a_{22}(r(t))\tanh(u_2(t))+b_{21}(r(t))\tanh(u_1(t-\tau_{ij}(t))) \\ \qquad +b_{22}(r(t))\tanh(u_2(t-\tau_{ij}(t))) \\ \qquad +c_{21}(r(t))\int_{-\infty}^{t} e^{t-s}\tanh(u_1(s))ds+c_{22}(r(t))\int_{-\infty}^{t} e^{t-s}\tanh(u_2(s))ds\} \end{cases},$$

$$t \ge 0, \quad (10)$$

where $J_i=0$, $i=1,2$. We assume that $\tau_{ij}(t)=1+0.1\sin t$, $i,j=1,2$. It is easy to verify that Assumption A1-A2 are satisfied, and we get $\gamma_1=\gamma_2=1$, $\underline{\theta}_1=0.5$, $\overline{\theta}_1=2.5$, $\underline{\theta}_2=1$, $\overline{\theta}_2=3$. We assume that $d_1^1=10$ $d_2^1=12$ $d_1^2=9$ $d_2^2=10$ $d_1^3=8$ $d_2^3=12$. When $i=1$, let $\omega_1^1=1$, $\omega_1^2=0.5$, $\omega_1^3=1$; when $i=2$, let $\omega_2^1=2$, $\omega_2^2=1$, $\omega_2^3=0.5$, and let jumping transfer parameters be $\pi_{11}=-1/2$, $\pi_{12}=1/8$, $\pi_{13}=3/8$, $\pi_{21}=1/6$, $\pi_{22}=-1/2$, $\pi_{23}=1/3$, $\pi_{31}=1/4$, $\pi_{11}=1/4$, $\pi_{11}=-1/2$.

We assume weighted matrix:

$$A(k=1)=\begin{bmatrix}-0.2 & 0.4\\ 0.3 & -0.5\end{bmatrix},\ A(k=2)=\begin{bmatrix}-0.4 & 0.8\\ 0.3 & -0.2\end{bmatrix},\ A(k=3)=\begin{bmatrix}-0.5 & 0.4\\ 0.4 & -0.5\end{bmatrix};$$

$$B(k=1)=\begin{bmatrix}-0.6 & 1\\ 0.5 & -1\end{bmatrix},\ B(k=2)=\begin{bmatrix}-0.2 & 1\\ 0.5 & -0.1\end{bmatrix},\ B(k=3)=\begin{bmatrix}-0.4 & 1\\ 0.5 & -0.5\end{bmatrix};$$

$$C(k=1)=\begin{bmatrix}-0.1, & 0.2\\ 0.1 & -0.3\end{bmatrix}, C(k=2)=\begin{bmatrix}-0.3 & 0\\ 0.5 & -0.4\end{bmatrix},\ C(k=3)=\begin{bmatrix}-0.2 & 0.1\\ 0.2 & -0.5\end{bmatrix}.$$

By calculation, we obtain

$$Q(1)=\begin{bmatrix}5.06 & -3.75\\ -9.3 & 12.325\end{bmatrix},\ Q(2)=\begin{bmatrix}3.83 & -3.5\\ -9.3 & 10\end{bmatrix},\ Q(3)=\begin{bmatrix}4.125 & -9\\ -4.65 & 11\end{bmatrix}.$$

Obviously, $Q(1)$, $Q(2)$ and $Q(3)$ are M matrix, so from Theorem 1 the equilibrium point $(0,0)^T$ of system (10) is stochastically exponentially stable.

5 Conclusion

In this paper, we have dealt with the problem of stochastic exponential stability of the equilibrium point of a class of delayed Cohen-Grossberg neural networks with Markovian jumping parameters by applying different analysis methods, *i.e.* vector Lyapunov function method and M-matrix theory. An example given at last illustrates the practicability of our results.

Acknowledgments. This work is supported by Natural Science Foundation of China (No. 60974132, No. 50823004), Doctoral Fund of Ministry of Education of China (No. 200806130003).

References

1. Arik, S., Tavanoglu, V.: On the Global Asymptotic Stability of Delayed Cellular Neural Networks. J. IEEE Transactions on Circuits and Systems—I 47, 571–574 (2000)
2. Wang, L., Zou, X.: Exponential Stability of Cohen-Grossberg Neural Networks. J. Neural Networks 15, 415–422 (2002)
3. Zhang, J., Suda, Y., Iwasa, T.: Absolutely Exponential Stability of Cohen-Grossberg Neural Networks with Variable Delays. J. Physics Letters A 338, 44–50 (2005)
4. Zhang, J., Suda, Y., Iwasa, T.: Absolutely Exponential Stability of a Class of Neural Networks with Unbounded Delayed. J. Neural Networks 17, 391–397 (2004)
5. Xiong, W., Cao, J.: Absolutely Exponential Stability of Cohen-Grossberg Neural Networks with Unbounded Delays. J. Neural Computing 68, 1–12 (2005)
6. Cao, J., Liang, J.: Boundedness and Stability for Cohen-Grossberg Neural Networks with Time-Varying Delays. J. Math. Anal. Appl. 296, 665–685 (2004)
7. Liu, H., Zhao, L., Zhang, Z., Ou, Y.: Stochastic Stability of Markovian Jumping Hopfield Neural Networks with Constant and Distributed Delays. J. Neurocomputing 72, 3669–3674 (2009)
8. Wang, L., Zhang, Z., Wang, Y.: Stochastic Exponential Stability of the Delayed Reaction-Diffusion Recurrent Neural Networks with Markovian Jumping Parameters. J. Physics Letters A 372, 3201–3209 (2008)
9. Sathy, R., Balasubramaniam, P.: Stability Analysis of Fuzzy Markovian Jumping Cohen-Grossberg BAM Neural Networks with Mixed Time-Varying Delays. J. Commun. Nonlinear Sci. Numer. Simulat. 16, 2054–2064 (2011)
10. Lou, X., Cui, B.: Stochastic Stability Analysis for Delayed Neural Networks of Neutral Type with Markovian Jump Parameters. J. Chaos, Solitons and Fractals 39, 2188–2197 (2009)
11. Lou, X., Cui, B.: Stochastic Exponential Stability for Markovian Jumping BAM Neural Networks with Time-Varying Delays. J. IEEE Transaction on System, Man, and Cybernetics-Part B Cybernetics 37, 713–719 (2007)
12. Balasubramaniam, P., Rakkiyappan, R.: Delay-Dependent Robust Stability Analysis for Markovian Jumping Stochastic Cohen-Grossberg Neural Networks with Discrete Interval and Distributed Time-Varying Delays. J. Nonlinear Analysis: Hybrid Systems 3, 207–214 (2009)
13. Cohen, M.A., Grossberg, S.: Absolute Stability and Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks. J. IEEE Trans. Syst. Man Cybern. SMC 13, 815–825 (1983)

Periodic Solutions for High-Order Cohen-Grossberg-Type BAM Neural Networks with Time-Delays

Yunquan Ke* and Chunfang Miao

Department of Mathematics, Shaoxing University, Shaoxing 312000,
P.R. China, Tel.: +86 57588341897

Abstract. In this paper, the existence and exponential stability is studied of periodic solutions for a class of high-order Cohen-Grossberg-type BAM neural networks with time-delays. By differential mean value theorem, integral mean value theorem and poincaré mapping, several sufficient conditions guaranteeing the existence, uniqueness and exponential stability of periodic solutions for high-order Cohen-Grossberg-type BAM neural networks with time-delays are given. An illustrative examples are also given in the end to show the effectiveness of our results.

Keywords: High-order Cohen-Grossberg-type BAM neural networks; Mean value theorem; Periodic solution; Exponential stability.

1 Introduction

Bidirectional associative memory neural networks and Cohen- Grossberg neural networks have been extensively studied in past years and found many applications in different fields such as signal processing, associative memory, combinatorial optimization and automatic control [1-5]. The Cohen-Grossberg-type BAM neural networks model (i.e., the BAM model which possesses Cohen-Grossberg dynamics, and is initially proposed by Cohen and Grossberg [6]), have great promising potential for the tasks of parallel computation, associative memory ect.. These applications heavily depend on the dynamical behaviors of system. Thus, the analysis of the dynamical behaviors of Cohen- Grossberg-type BAM neural networks are important and necessary. In recent years, many researchers have studied the global stability and other dynamical behaviors of the Cohen-Grossberg-type BAM neural networks, see [7-11]. For example, In [7], Cao and Song investigated the global exponential stability for Cohen-Grossberg-type BAM neural networks with time-varying delays by using Lyapunov function, M-matrix theory and inequality technique. In [10], by constructing a suitable Lyapunov functional, the asymptotic stability was investigated for Cohen-Grossberg-type BAM neural network. In [11], the authors have proposed a new Cohen-Grossberg-type BAM neural network model with time

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

delays, and some new sufficient conditions ensuring the existence and global asymptotical stability of equilibrium point for this model have been derived.

In addition, the research of neural networks with delays involves not only the dynamic analysis of equilibrium point but also that of periodic oscillatory solution. In practice, the dynamic behavior of periodic oscillatory solution is very important in learning theory. Moreover, it is well known that an equilibrium point can be viewed as a special periodic solution of neural networks with arbitrary period. In this sense, the analysis of periodic solutions of neural networks with delays to be more general than that of equilibrium point. Several important results for periodic solutions of neural networks with delays have been obtained in Refs. [12-15]. For example, The authors in [13,14,15] have investigated the periodicity of delayed Cohen-Grossberg-type BAM neural networks with variable coefficients. The authors in [16] have investigated the periodic solutions for a class of higher-order Cohen-Grossberg type neural networks with delays. In [17], Huo , Wan and Sanyi have investigated the dynamics of high-order BAM neural networks with and without impulses. The authors have investigated the exponential stability of high-order neural networks with time delays [18-19].

Motivated by the above discussions, a class of high-order Cohen-Grossberg-type BAM neural networks with time delay is considered in this paper. We will derive some sufficient conditions of existence, uniqueness and exponential stability of periodic solution for high-order Cohen-Grossberg-type BAM neutral networks with time delays by constructing suitable Lyapunov function, using differential mean value theorem , integral mean value theorem and poincar$\acute{e}$ mapping.

Consider the following high-order Cohen-Grossberg-type BAM neural networks

$$\begin{cases} \frac{du_i(t)}{dt} = -a_i(u_i(t))[b_i(u_i(t)) - \sum\limits_{j=1}^{m} c_{ij} f_j(v_j(t-\tau_{ij})) \\ \qquad - \sum\limits_{j=1}^{m}\sum\limits_{k=1}^{m} T_{ijk} f_j(v_j(t)) f_k(v_k(t)) - I_i(t)], \\ \frac{dv_j(t)}{dt} = -d_j(v_j(t))[e_j(v_j(t)) - \sum\limits_{i=1}^{n} h_{ji} g_i(u_i(t-\sigma_{ji})) \\ \qquad - \sum\limits_{i=1}^{n}\sum\limits_{q=1}^{n} S_{jiq} g_i(u_i(t)) g_q(u_q(t)) - J_j(t)], \end{cases} \tag{1}$$

for $i = 1, 2, \cdots, n, j = 1, 2, \cdots, m$, where $u_i(t)$ and $v_j(t)$ are the states of the ith neuron from the neural field F_U and the jth neuron from the neural field F_V at the time t, respectively; f_j, g_i denote the activation functions of jth neuron from F_V and the ith neuron from F_U, respectively; c_{ij} weights the strength of the ith neuron on the jth neuron at the time $t - \tau_{ij}$; h_{ji} weights the strength of the jth neuron on the ith neuron at the time $t - \sigma_{ji}$; $\tau_{ij} \geq 0$ and $\sigma_{ji} \geq 0$; $I_i(t), J_j(t)$ denote the external inputs on the ith neuron from F_U and the jth neuron from F_V at the time t , respectively; $a_i(u_i(t))$ and $d_j(v_j(t))$ represent amplification functions; $b_i(u_i(t))$ and $e_j(v_j(t))$ are appropriately behaved functions such that the solutions of model(1) remain bounded; T_{ijk} and S_{jiq} are constants, and denote the first- and second-order connection weights of the neural networks, respectively.

The initial conditions of system (1) are given by

$$\begin{cases} u_i(s) = \phi_{ui}(s), s \in [-\sigma, 0], \\ v_j(s) = \phi_{vj}(s), s \in [-\tau, 0], \end{cases} \tag{2}$$

where $i = 1, 2, \ldots .n, j = 1, 2, \ldots .m$, $\sigma = \max\limits_{1 \le i \le n, 1 \le j \le m} \{\sigma_{ji}\}$, $\tau = \max\limits_{1 \le i \le n, 1 \le j \le m} \{\tau_{ij}\}$, $\phi_{ui}(s)$ and $\phi_{vj}(s)$ are bounded and continuous on $[-\delta, 0]$, $\delta = \max\{\sigma, \tau\}$.

2 Preliminaries

In order to establish the existence, uniqueness and exponential stability of periodic solution for system (1), we give some assumptions.
$(H1)$: For each $i = 1, 2, \ldots, n; j = 1, 2, \ldots, m$, functions $a_i(x), d_j(x))$ are continuously bounded and satisfy $0 < \underline{a}_i \le a_i(x) \le \bar{a}_i, \quad 0 < \underline{d}_j \le d_j(x) \le \bar{d}_j$, for all $x \in R$.
$(H2)$: There exist $\beta_i > 0$ and $\gamma_j > 0$, $i = 1, 2, \ldots, n; j = 1, 2, \ldots, m$, such that

$$(x-y)[b_i(x)-b_i(y)] \ge \beta_i(x-y)^2, \ \ (x-y)[e_j(x)-e_j(y)] \ge \gamma_j(x-y)^2, \quad \forall x, y \in R.$$

$(H3)$: The activation functions f_j and $g_i (i = 1, 2, \ldots, n; j = 1, 2, \ldots, m)$ satisfy Lipschitz condition, that is, there exist constant $F_j > 0$ and $G_i > 0$, such that

$$|f_j(\xi_1) - f_j(\xi_2)| \le F_j|\xi_1 - \xi_2|, \quad |g_i(\xi_1) - g_i(\xi_2)| \le G_i|\xi_1 - \xi_2|, \ \forall \xi_1, \xi_2 \in R.$$

$(H4)$: There exist numbers $N_i > 0$ and $M_j > 0$ such that

$$|f_j(x)| \le M_j, \quad |g_i(x)| \le N_i, \quad \forall x \in R.$$

$(H5)$: The activation functions $f_j(x)$ and $g_i(x)(i = 1, 2, \ldots, n; j = 1, 2, \ldots, m)$ are continuously differentiable on x, $x \in R$.
$(H6)$: The activation functions $f_j(x)$ and $g_i(x)(i = 1, 2, \ldots, n; j = 1, 2, \ldots, m)$ are continuously differentiable on x, and there exist $L_j > 0$ and $T_i > 0$, such that $|\frac{df_j(x)}{dx}| \le L_j, \quad |\frac{dg_i(x)}{dx}| \le T_i, \quad \forall x \in R$.
$(H7)$: $I_i(t)$ and $J_j(t)$ are continuously periodic functions defined on $t \in [0, \infty)$ with common period $\omega > 0$, and they are all bounded, denote

$$I_i^* = \sup_{0 \le t < \infty} |I_i(t)|, \quad J_j^* = \sup_{0 \le t < \infty} |J_j(t)|, \ \ i = 1, 2, \cdots, n, j = 1, 2, \cdots, m.$$

Definition 2.1. For $u(t) = (u_1(t), u_2(t), \cdots, u_n(t))^T, \quad v(t) = (v_1(t), v_2(t), \cdots, v_m(t))^T$, we define the norm: $\|u\|^2 = \sum\limits_{i=1}^{n} |u_i(t)|^2, \quad \|v\|^2 = \sum\limits_{j=1}^{m} |v_i(t)|^2$.

Definition 2.2. Let $Z^*(t) = (u_1^*(t), u_2^*(t), \cdots, u_n^*(t), v_1^*(t), v_2^*(t), \cdots, v_m^*(t))^T$ be an $\omega-$ periodic solution of system (1) with initial value

$$\psi = (\psi_{u1}(t), \psi_{u2}(t), \cdots, \psi_{un}(t), \psi_{v1}(t), \psi_{v2}(t), \cdots, \psi_{vm}(t))^T.$$

If there exist constants $\alpha > 0$ and $M > 1$, for every solution $Z(t) = (u_1(t), u_2(t), \cdots, u_n(t), v_1(t), v_2(t), \cdots, v_m(t))^T$ of system (1) with initial value

$$\phi = (\phi_{u1}(t), \phi_{u2}(t), \cdots, \phi_{un}(t), \phi_{v1}(t), \phi_{v2}(t), \cdots, \phi_{vm}(t))^T,$$

such that

$$\sum_{i=1}^{n} |u_i(t) - u_i^*(t)|^2 + \sum_{j=1}^{m} |v_j(t) - v_j^*(t)|^2 \leq Me^{-\alpha t}[\|\phi_u - \psi_u\|^2 + \|\phi_v - \psi_v\|^2],$$

then $Z^*(t)$ is said to be exponentially stable, where

$$\|\phi_u - \psi_u\|^2 = \sup_{-\sigma \leq t \leq 0} \sum_{i=1}^{n} |\phi_{ui}(t) - \psi_{ui}(t)|^2, \ \|\phi_v - \psi_v\|^2 = \sup_{-\tau \leq t \leq 0} \sum_{j=1}^{m} |\phi_{vj}(t) - \psi_{vj}|^2.$$

Lemma 2.1. Under hypotheses $(H1), (H2), H(4)$ and $(H7)$, there exist constants $R_i > 0$ and $R_j^* > 0$ such that

$$|u_i(t)| \leq R_i, \quad |v_j(t)| \leq R_j^*, \ \ i = 1, 2, \cdots, n, j = 1, 2, \cdots, m, \ \ t > 0.$$

Lemma 2.2. For $(x_1(t), x_2(t), \cdots, x_m(t))^T, (x_1^*(t), x_2^*(t), \cdots, x_m^*(t))^T \in R^m$, if $h_j(x_j)$ are continuously differentiable on $x_j (j = 1, 2, \ldots, m)$, then we have
(A1) $\sum\limits_{j=1}^{m} \sum\limits_{k=1}^{m} b_{ijk}[h_j(x_j)h_k(x_k) - h_j(x_j^*)h_k(x_k^*)] = \sum\limits_{j=1}^{m} \sum\limits_{k=1}^{m} (b_{ijk} + b_{ikj}) \frac{\partial h_j(\xi_j)}{\partial x_j}(x_j - x_j^*)h_k(\xi_k).$
Or,
(A2) $\sum\limits_{j=1}^{m} \sum\limits_{k=1}^{m} b_{ijk}[h_j(x_j)h_k(x_k) - h_j(x_j^*)h_k(x_k^*)] = \sum\limits_{j=1}^{m} \sum\limits_{k=1}^{m} (b_{ijk} + b_{ikj})[h_j(x_j) - h_j(x_j^*)]h_k(\xi_k),$
where ξ_j lies between x_j and x_j^*, ξ_k lies between x_k and x_k^*, $j, k = 1, 2, \ldots, m$.
Lemma 2.3. Assume that

$$- 2\underline{a}_i\beta_i + \bar{a}_i \sum_{j=1}^{m}[|c_{ij}| + \sum_{k=1}^{m} |T_{ijk} + T_{ikj}|M_k]F_j + \sum_{j=1}^{m}[|h_{ji}| + \sum_{q=1}^{n} |S_{jiq} + S_{jqi}|N_q]G_i\bar{d}_j < 0, \tag{3}$$

$$- 2\underline{d}_j\gamma_j + \sum_{i=1}^{n}[|c_{ij}| + \sum_{k=1}^{m} |T_{ijk} + T_{ikj}|M_K]\bar{a}_iF_j + \bar{d}_j \sum_{i=1}^{n}[|h_{ij}| + \sum_{q=1}^{n} |S_{jiq} + S_{jqi}|N_q]G_i < 0, \tag{4}$$

for $i = 1, 2, \ldots. n, j = 1, 2, \ldots. m$, then there exists $\alpha > 0$, such that

$$\alpha - 2\underline{a}_i\beta_i + \bar{a}_i \sum_{j=1}^{m} [|c_{ij}| + \sum_{k=1}^{m} |T_{ijk} + T_{ikj}|M_k]F_j + \sum_{j=1}^{m} [e^{\alpha\sigma}|h_{ji}| + \sum_{q=1}^{n} |S_{jiq} + S_{jqi}|N_q]G_i\bar{d}_j \leq 0,$$

$$\alpha - 2\underline{d}_j\gamma_j + \sum_{i=1}^{n}[e^{\alpha\tau}|c_{ij}| + \sum_{k=1}^{m}|T_{ijk} + T_{ikj}|M_K]\bar{a}_iF_j$$
$$+ \bar{d}_j\sum_{i=1}^{n}[|h_{ij}| + \sum_{q=1}^{n}|S_{jiq} + S_{jqi}|N_q]G_i \le 0,$$

for $i = 1, 2, \ldots .n, j = 1, 2, \ldots .m.$
They are clear that the result of Lemma 2.1 and Lemma 2.3 are correct, is omitted of proof.

3 Main Results

Theorem 3.1. For the system (1), under the hypotheses $(H1)-(H5)$ and $(H7)$, there exists one ω-periodic solution of system (1), and all other solutions of (1) exponentially converge to it as $t \to +\infty$, if (3) and (4) in Lemma 2.3 hold.

Proof. Let

$$\Omega = \{\phi|\phi = \begin{pmatrix}\phi_u^T\\ \phi_v^T\end{pmatrix}, \phi_u = (\phi_{u1}, \phi_{u2}, \cdots, \phi_{un})^T, \phi_v = (\phi_{v1}, \phi_{v2}, \cdots, \phi_{vm})^T\}.$$

For any $\phi = (\phi_u^T, \phi_v^T)^T \in \Omega$, we define the norm of ϕ: $\|\phi\| = \|\phi_u\| + \|\phi_v\|$, in which $\|\phi_u\|^2 = \sup\limits_{-\sigma\le s\le 0}\sum\limits_{i=1}^{n}|\phi_{ui}(s)|^2, \quad \|\phi_v\|^2 = \sup\limits_{-\tau\le s\le 0}\sum\limits_{j=1}^{m}|\phi_{vj}(s)|^2,$
then Ω is the Banach space of continuous functions which map $([-\sigma, 0], [-\tau, 0])^T$ into R^{n+m} with the topology of uniform convergence. For any $(\phi_u^T, \phi_v^T)^T, (\psi_u^T, \psi_v^T)^T \in \Omega$, we denote the solutions of system (1) in the initial conditions

$$\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}\phi_u^T\\ \phi_v^T\end{pmatrix}\right), \left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}\psi_u^T\\ \psi_v^T\end{pmatrix}\right),$$

as $u(t, \phi_u) = (u_1(t, \phi_u), u_2(t, \phi_u), \cdots, u_n(t, \phi_u))^T$, $v(t, \phi_v) = (v_1(t, \phi_v), v_2(t, \phi_v), \cdots, v_m(t, \phi_v))^T$, and
$u(t, \psi_u) = (u_1(t, \psi_u), u_2(t, \psi_u), \cdots, u_n(t, \psi_u))^T, v(t, \psi_v) = (v_1(t, \psi_v), v_2(t, \psi_v), \cdots, v_m(t, \psi_v))^T$, respectively. Defining

$$u_t(\phi_u) = u(t + \rho, \phi_u), \rho \in [-\sigma, 0], \quad v_t(\phi_v) = v(t + \rho, \phi_v), \rho \in [-\tau, 0], t > 0,$$

then $(u_t(\phi_u), v_t(\phi_v))^T \in \Omega$, for all $t > 0$.
Let $y_i(t) = u_i(t, \phi_u) - u_i(t, \psi_u), \quad z_j(t) = v_j(t, \phi_v) - v_j(t, \psi_v),$

$$\bar{y}_i(t) = \int_{u_i(t,\psi_u)}^{u_i(t,\phi_u)} \frac{ds}{a_i(s)}, \quad \bar{z}_j(t) = \int_{v_j(t,\psi_v)}^{v_j(t,\phi_v)} \frac{ds}{d_j(s)}, i = 1, 2, \cdots, n, j = 1, 2, \cdots, m.$$

Note that $a_i(s), d_j(s)$ are continuous, $a_i(s) > 0, d_j(s) > 0$, from Lemma 2.1, $u_i(t, \phi_u), v_j(t, \phi_v)$, $u_i(t, \psi_u)$ and $v_j(t, \psi_v)$ are bounded. By mean-value theorem for integral, we have $\bar{y}_i(t) = \frac{1}{a_i(\xi_i)}[u_i(t, \phi_u) - u_i(t, \psi_u)] = \frac{1}{a_i(\xi_i)}y_i(t)$, where $\xi_i \in [\min\{u_i(t, \psi_u), u_i(t, \phi_u)\}, \max\{u_i(t, \psi_u), u_i(t, \phi_u)\}]$, and

$$\bar{z}_j(t) = \frac{1}{d_j(\eta_j)}[v_j(t, \phi_v) - v_j(t, \psi_v)] = \frac{1}{d_j(\eta_j)}z_j(t),$$

where $\eta_j \in [\min\{v_j(t,\psi_v), v_j(t,\phi_v)\}, \max\{v_j(t,\psi_v), v_j(t,\phi_v)\}]$, then we have
$sgn(\bar{y}_i(t)) = sgn(y_i(t)), sgn(\bar{z}_j(t)) = sgn(z_j(t))$.

From (1), and (A2) of Lemma 2.2, we derive

$$\frac{d|\bar{y}_i(t)|}{dt} = sgn(y_i(t))\{-[b_i(u_i(t,\phi_u)) - b_i(u_i(t,\psi_u))] + \sum_{j=1}^{m} c_{ij}[f_j(v_j(t-\tau_{ij},\phi_v)) - f_j(v_j(t-\tau_{ij},\psi_v))] + \sum_{j=1}^{m}\sum_{k=1}^{m}(T_{ijk}+T_{ikj})[f_j(v_j(t,\phi_v)) - f_j(v_j(t,\psi_v))].f_k(v_j(t,\psi_v) + (v_j(t,\phi_v) - v_j(t,\psi_v))\theta_1)\}, \quad (5)$$

$$\frac{d|\bar{z}_j(t)|}{dt} = sgn(z_j(t))\{-[d_j(v_j(t,\phi_v)) - d_j(v_j(t,\psi_v))] + \sum_{i=1}^{n} h_{ji}[g_i(u_i(t-\sigma_{ji},\phi_u)) - g_i(u_i(t-\sigma_{ji},\psi_u))] + \sum_{i=1}^{n}\sum_{q=1}^{n}(S_{jiq}+S_{jqi})[g_i(u_i(t,\phi_u)) - g_i(u_i(t,\psi_u))].g_q(u_i(t,\psi_u) + (u_i(t,\phi_u) - u_i(t,\psi_u))\theta_2)\}, \quad (6)$$

for $i = 1,2,\cdots,n, j = 1,2,\cdots,m, 0 < \theta_1,\theta_2 < 1$.

We consider the following Lyapunov function

$$V(t) = e^{\alpha t}\sum_{i=1}^{n} a_i^2(\xi_i)|\bar{y}_i(t)|^2 + \sum_{i=1}^{n}\sum_{j=1}^{m}|c_{ij}|\bar{a}_i F_j \int_{t-\tau_{ij}}^{t} e^{\alpha(s+\tau_{ij})}|z_j(s)|^2 ds + e^{\alpha t}\sum_{j=1}^{m} d_j^2(\eta_j)|\bar{z}_j(t)|^2 + \sum_{j=1}^{m}\sum_{i=1}^{n}|h_{ji}|\bar{d}_j G_i \int_{t-\sigma_{ji}}^{t} e^{\alpha(s+\sigma_{ji})}|y_i(s)|^2 ds, \quad (7)$$

where α is given by Lemma 2.1.

Calculate the rate of change of $V(t)$ along (5)-(6), we derive

$$D^+V(t) \le e^{\alpha t}\sum_{i=1}^{n}\{\alpha - 2\underline{a}_i\beta_i + \bar{a}_i\sum_{j=1}^{m}[|c_{ij}| + \sum_{k=1}^{m}|T_{ijk}+T_{ikj}|M_k]F_j + \sum_{j=1}^{m}[e^{\alpha\sigma}|h_{ji}| + \sum_{q=1}^{n}|S_{jiq}+S_{jqi}|N_q]\bar{d}_j G_i\}y_i^2(t) + e^{\alpha t}\sum_{j=1}^{m}\{\alpha - 2\underline{d}_j\gamma_j + \sum_{i=1}^{n}[e^{\alpha\tau}|c_{ij}| + \sum_{k=1}^{m}|T_{ijk}+T_{ikj}|M_k]\bar{a}_i F_j + \sum_{i=1}^{n}[|h_{ji}| + \sum_{q=1}^{n}|S_{jiq}+S_{jqi}|N_q]G_i\bar{d}_j\}z_j^2(t). \quad (8)$$

By Lemma 2.3, from (8), we can find $D^+V(t) \le 0$, and so $V(t) \le V(0)$, for all $t > 0$. From (7), we have

$$V(t) \ge e^{\alpha t}[\sum_{i=1}^{n}|y_i(t)|^2 + \sum_{j=1}^{m}|z_j(t)|^2], \quad t \ge 0. \quad (9)$$

$$V(0) = \sum_{i=1}^{n}|y_i(0)|^2 + \sum_{j=1}^{m}|z_j(0)|^2 + \sum_{i=1}^{n}\sum_{j=1}^{m}|c_{ij}|\bar{a}_i F_j \int_{-\tau_{ij}}^{0} e^{\alpha(s+\tau_{ij})}|z_j(s)|^2 ds + \sum_{j=1}^{m}\sum_{i=1}^{n}|h_{ji}|\bar{d}_j G_i \int_{-\sigma_{ji}}^{0} e^{\alpha(s+\sigma_{ji})}|y_i(s)|^2 ds. \quad (10)$$

From (8)-(10), we obtain
$\sum_{i=1}^{n}|u_i(t,\phi_u)-u_i(t,\psi_u)|^2+\sum_{j=1}^{m}|v_j(t,\phi_v)-v_j(t,\psi_v)|^2$
$\le e^{-\alpha t}[1+\frac{e^{\alpha\sigma}}{\alpha}\sum_{j=1}^{m}\max_{1\le i\le n}(|h_{ji}|G_i)\bar{d}_j]\|\phi_u-\psi_u\|^2+e^{-\alpha t}[1+$

$$\frac{e^{\alpha\tau}}{\alpha}\sum_{i=1}^{n}\max_{1\le j\le m}(|c_{ij}|F_j)\bar{a}_i]\|\phi_v-\psi_v\|^2=Me^{-\alpha t}[\|\phi_u-\psi_u\|^2+\|\phi_v-\psi_v\|^2],\quad (11)$$

where $M=\max\{1+\frac{e^{\alpha\sigma}}{\alpha}\sum_{j=1}^{m}\max_{1\le i\le n}(|h_{ji}|G_i)\bar{d}_j,1+\frac{e^{\alpha\tau}}{\alpha}\sum_{i=1}^{n}\max_{1\le j\le m}(|c_{ij}|F_j)\bar{a}_i\}>1$.

Let $|u(\phi_u)-u(\psi_u)|^2=\sum_{i=1}^{n}|u_i(t,\phi_u)-u_i(t,\psi_u)|^2, |u(\phi_v)-u(\psi_v)|^2=\sum_{j=1}^{m}|v_j(t,\phi_v)-v_j(t,\psi_v)|^2$.
From (13), we have $|u(\phi_u)-u(\psi_u)|^2\le Me^{-\alpha t}[\|\phi_u-\psi_u\|^2+\|\phi_v-\psi_v\|^2],\ \ t>0,$
$|u(\phi_v)-u(\psi_v)|^2\le Me^{-\alpha t}[\|\phi_u-\psi_u\|^2+\|\phi_v-\psi_v\|^2],\quad t>0.$

We can choose a positive integer N, such that $Me^{-\alpha(N\omega+\rho)}\le\frac{1}{3},\rho\in[-\delta,0]$. Now we define a Poincaré mapping F: $\Omega\rightarrow\Omega$ by $F(\phi_u^T,\phi_v^T)^T=(u_\omega^T(\phi_u),v_\omega^T(\phi_v))^T$, then $F^N(\phi_u^T,\phi_v^T)^T=(u_{N\omega}^T(\phi_u),v_{N\omega}^T(\phi_v))^T$. Let $t=N\omega$, then have
$|F^N\phi_u-F^N\psi_u|^2\le\frac{1}{3}[\|\phi_u-\psi_u\|^2+\|\phi_v-\psi_v\|^2],\quad |F^N\phi_v-F^N\psi_v|^2\le\frac{1}{3}[\|\phi_u-\psi_u\|^2+\|\phi_v-\psi_v\|^2]$.
This implies that F^N is a contraction mapping, hence there exist a unique fixed point $(\phi_u^{*T},\phi_v^{*T})^T\in\Omega$, such that $F^N(\phi_u^{*T},\phi_v^{*T})^T=(\phi_u^{*T},\phi_v^{*T})^T$. Since

$$F^N\left(F\begin{pmatrix}\phi_u^{*T}\\ \phi_v^{*T}\end{pmatrix}\right)=F\left(F^N\begin{pmatrix}\phi_u^{*T}\\ \phi_v^{*T}\end{pmatrix}\right)=F\begin{pmatrix}\phi_u^{*T}\\ \phi_v^{*T}\end{pmatrix},$$

then $F(\phi_u^{*T},\phi_v^{*T})^T\in\Omega$ is also a fixed point of F^N, and so $F(\phi_u^{*T},\phi_v^{*T})^T=(\phi_u^{*T},\phi_v^{*T})^T$, i.e., $(u_\omega(\phi_u^{*T}),v_\omega(\phi_v^{*T}))^T=(\phi_u^{*T},\phi_v^{*T})^T$. Let $(u(t,\phi_u^{*T}),v(t,\phi_v^{*T}))^T$ be the solution of system (1) through $((0,0)^T,(\phi_u^{*T},\phi_v^{*T}))$, then $(u^T(t+\omega,\phi_u^*),v^T(t+\omega,\phi_v^*))^T$ is also a solution of (1). Obviously

$$\begin{pmatrix}u_{t+\omega}^T(\phi_u^*)\\ v_{t+\omega}^T(\phi_v^*)\end{pmatrix}=\begin{pmatrix}u_t^T(u_\omega(\phi_u^*))\\ v_t^T(v_\omega(\phi_v^*))\end{pmatrix}=\begin{pmatrix}u_t^T(\phi_u^*)\\ v_t^T(\phi_v^*)\end{pmatrix},$$

for all $t>0$. Hence

$$\begin{pmatrix}u^T(t+\omega,\phi_u^*)\\ v^T(t+\omega,\phi_v^*)\end{pmatrix}=\begin{pmatrix}u^T(t,\phi_u^*)\\ v^T(t,\phi_v^*)\end{pmatrix},$$

for all $t>0$.

This shows which is exactly one $\omega-$ periodic solution of system (1) and other solutions of system (1) exponentially converge to it as $t\rightarrow+\infty$. □

On the one hand, by using (A1) of Lemma 2.2, the proof is similar to that in proving Theorem 3.1, we may obtain the following Theorem 3.2 results. Here we omit the proof.

Theorem 3.2. Under hypotheses $(H1)-(H4)$, $(H6)$ and $(H7)$, there exists one ω-periodic solution of system (1), and all other solutions of (1) exponentially converge to it as $t \to +\infty$, if

$$-2\underline{a}_i\beta_i + \bar{a}_i \sum_{j=1}^{m}[|c_{ij}|F_j + \sum_{k=1}^{m}|T_{ijk}+T_{ikj}|M_kL_j] + \sum_{j=1}^{m}[|h_{ji}|G_i + \sum_{q=1}^{n}|s_{jiq}+s_{jqi}|N_qT_i]\bar{d}_j < 0, \tag{12}$$

$$-2\underline{d}_j\gamma_j + \sum_{i=1}^{n}[|c_{ij}|F_j + \sum_{k=1}^{m}|T_{ijk}+T_{ikj}|M_kL_j]\bar{a}_i + \bar{d}_j\sum_{i=1}^{n}[|h_{ji}|G_i + \sum_{q=1}^{n}|s_{jiq}+s_{jqi}|N_qT_i] < 0, \tag{13}$$

for $i = 1, 2, \ldots .n, j = 1, 2, \ldots .m$.

Remark 1. Theorem 3.1 and Theorem 3.2 are developed under different assumptions and use of various lemmas. They provide different sufficient conditions ensuring the periodic solution of system (1) to be exponentially stable. Therefore, we can select suitable theorems for a high-order Cohen-Grossberg-type BAM neural networks to determine the existence and exponential stability of the periodic solution.

4 Examples

Example 4.1. Consider the following high-order Cohen-Grossberg-type BAM neural networks$(n = m = 2)$

$$\begin{cases} \frac{du_i(t)}{dt} = -a_i(u_i(t))[b_i(u_i(t)) - \sum_{j=1}^{2} c_{ij}f_j(v_j(t-\tau_{ij})) \\ \qquad - \sum_{j=1}^{2}\sum_{k=1}^{2} T_{ijk}f_j(v_j(t))f_k(v_k(t)) - I_i(t)], \\ \frac{dv_j(t)}{dt} = -d_j(v_j(t))[e_j(v_j(t)) - \sum_{i=1}^{2} h_{ji}g_i(u_i(t-\sigma_{ji})) \\ \qquad - \sum_{i=1}^{2}\sum_{q=1}^{2} S_{jiq}g_i(u_i(t))g_q(u_q(t)) - J_j(t)], \end{cases} \tag{14}$$

for $i = 1, 2, j = 1, 2$, where

$f_i(r) = g_i(r) = \sin\frac{r}{2}$, $a_i(r) = 3 + sinr$, $d_j(r) = 3 + cosr$, $b_i(r) = 9r$, $e_j(r) = 27r$, $I_i(r) = -\frac{1}{4}cosr$, $J_j(r) = sinr$, $i, j = 1, 2$. Since $\forall\, r_1, r_2 \in R$, $|f_i(r_1) - f_i(r_2)| = |g_i(r_1) - g_i(r_2)| \le \frac{1}{4}|r_1 - r_2|$, $|f_i(r)| = |g_i(r)| \le 1$, $2 \le a_i(r) \le 3$, $2 \le d_j(r) \le 3$, $b_i(r_1) - b_i(r_2) = 9(r_1 - r_2)$, $e_j(r_1) - d_i(r_2) = 27(r_1 - r_2)$.

$$\left|\frac{df_i(r)}{dr}\right| = \left|\frac{dg_i(r)}{dr}\right| = \frac{1}{2}\left|\cos\frac{r}{2}\right| \le \frac{1}{2}, \quad i = 1, 2.$$

We select $F_i = G_i = \frac{1}{4}$, $N_i = M_i = 1$, $L_i = T_i = \frac{1}{2}$, $\bar{a}_i = 4$, $\underline{a}_i = 2$, $\bar{d}_i = 4$, $\underline{d}_i = 2$, $\beta_i = 9$, $\gamma_i = 27$, $\tau_{ij} = 4$, $\sigma_{ij} = 5$, $i = 1, 2$.
Let $c_{11} = 1$, $c_{12} = -1$, $c_{21} = \frac{1}{2}$, $c_{22} = \frac{1}{2}$, $h_{11} = -1$, $h_{12} = 1$, $h_{21} = 1$, $h_{22} = -1, s_{111} = 1$, $s_{121} = -1$, $s_{211} = -1$, $s_{221} = -1$, $s_{112} = -1$, $s_{122} = -\frac{1}{2}$, $s_{212} = 1$, $s_{222} = 1, T_{111} = 1$, $T_{121} = -1$, $T_{211} = -1$, $T_{221} = 1$, $T_{112} = 1$, $T_{122} = -1$, $T_{212} = -1$, $T_{222} = -1$.

By calculation, we have (3) and (4) in Theorem 3.1 hold. It follows from Theorem 3.1 that this system has one unique 2π- periodic solution, and all other solutions of system exponentially converge to it as $t \to +\infty$. Figs. 1-4 depict the time responses of state variables of $u_1(t), u_2(t), v_1(t)$ and $v_2(t)$ of system in example 4.1, respectively.

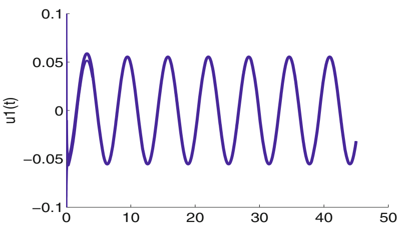


Fig. 1. Transient response of state variable u1(t)

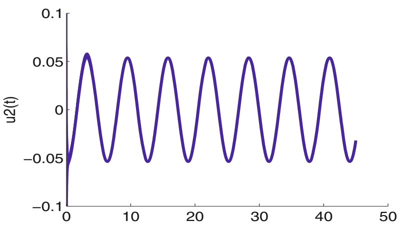


Fig. 2. Transient response of state variable u2(t)

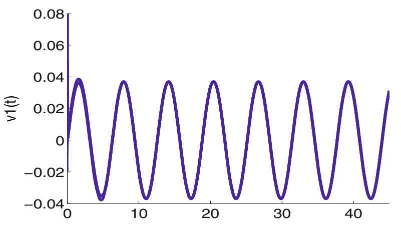


Fig. 3. Transient response of state variable v1(t)

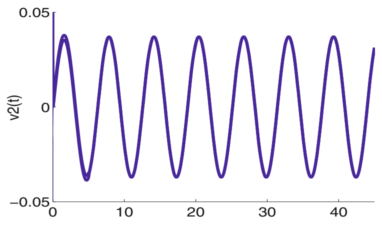


Fig. 4. Transient response of state variable v2(t)

Acknowledgments. This work is supported by the National Natural Science Foundation of Zhejiang Province (No.Y6100096), and the National Natural Science Foundation of China(No.10871226, No.10875078).

References

1. Kosko, B.: Adaptive bidirectional associative memories. Applied Optics 26, 4947–4960 (1987)
2. Kosko, B.: Bi-directional associative memories. IEEE Transactions on Systems. Man and Cybernetics 18, 49–60 (1988)
3. Mao, Z.: Dynamical analysis of Cohen-Grossberg neural networks with distributed delays. Physics Letters A 364, 38–47 (2007)

4. Bai, C.: Global exponential stability and existence of periodic solution of Cohen-Grossberg type neural networks with delays and impulses. Nonlinear Analysis: Real World Applications 9, 747–761 (2008)
5. Arik, S., Orman, Z.: Global stability analysis of Cohen-Grossberg neural networks with time varying delays. Physics Letters A 341, 410–421 (2005)
6. Cohen, M., Grossberg, S.: Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. IEEE Trans. Syst., Man Cybernetics 13, 815–826 (1983)
7. Cao, J., Song, Q.: Stability in Cohen-Grossberg-type bidirectional associative memory neural networks with time-varying delays. Nonlinearity 19, 1601–1617 (2006)
8. Yang, F., Zhang, C., Wu, D.: Global stability analysis of impulsive BAM type Cohen-Grossberg neural networks with delays. Appl. Math. Comput. 186, 932–940 (2007)
9. Zhou, Q., Wan, L.: Impulsive effects on stability of Cohen-Grossberg-type bidirectional associative memory neural networks with delays. Nonlinear Anal.: Real World Appl. 10, 2531–2540 (2009)
10. Bai, C.: Stability analysis of Cohen-Crossberg BAM neural networks with delays and impulses. Chaos, Solitons Fractals 35, 263–267 (2008)
11. Feng, C., Plamondon, P.: Stability analysis of bidirectional associative memory networks with time delays. IEEE Trans. Neural Networks 14, 1560–1565 (2003)
12. Jiang, H., Cao, J.: BAM-type Cohen-Grossberg neural networks with time delays. Math. Comput. Modell. 47, 92–103 (2008)
13. Chen, A., Cao, J.: Periodic bi-directional Cohen-Grossberg neural networks with distributed delays. Nonlinear Anal. 66, 2947–2961 (2007)
14. Li, Y., Fan, X.: Existence and globally exponential stability of almost periodic solution for Cohen-Grossberg BAM neural networks with variable coefficients. Appl. Math. Modell. 33, 2114–2120 (2009)
15. Xiang, H., Cao, J.: Exponential stability of periodic solution to Cohen-Grossberg-type BAM networks with time-varying delays. Neurocomputing 72, 1702–1711 (2009)
16. Ren, F.I., Cao, J.: Periodic solutions for a class of higher-order Cohen-Grossberg type neural networks with delays. Computers and Mathematics with Applications 54, 826–839 (2007)
17. Huo, H.F., Wan, T.L., Tang, S.: Dynamics of high-order BAM neural networks with and without impulses. Applied Mathematics and Computation 215, 2120–2133 (2009)
18. Cao, J., Liang, J., Lam, J.: Exponential stability of high-order bidirectional associative memory neural networks with time delays. Physics D 199, 425–436 (2004)
19. Zhang, J., Gui, Z.: Existence and stability of periodic solutions of high-order Hopfield neural networks with impulses and delays. J. Comput. Appl. Math. 224, 602–613 (2009)
20. 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 42, 354–366 (1995)

Simulation and Verification of Zhang Neural Networks and Gradient Neural Networks for Time-Varying Stein Equation Solving

Chenfu Yi[1], Yuhuan Chen[2], and Huajin Wang[1]

[1] School of Information Engineering, Jiangxi University of Science and Technology
[2] Center for Educational Technology, Gannan Normal University
Ganzhou 341000, China
Itchenve@gmail.com

Abstract. Differing from gradient-based neural networks (GNN), In this paper, we present a special kind of recurrent neural networks using a new design method to solve online the time-varying Stein matrix equation $A(t)X(t)B(t) + X(t) = C(t)$. This paper investigates simulation and verification of the resultant Zhang neural networks (ZNN) for the nonstationary Stein equation by using MATLAB simulation techniques. Theoretical analysis and simulation results substantiate the superior performance of the ZNN models for the solution of time-varying Stein equation in real-time, in compared with the GNN models.

Keywords: Neural networks, Time-varying, Stein equation, Global exponential convergence.

1 Introduction

The problem of Stein matrix equations solving is widely encountered in science and engineering, since it is usually an essential part in many applications such as control system design [1], optimization [2], and signal processing [3]. In view of these, we consider in this paper the following Stein matrix equation (which could also be viewed as a linear matrix equation):

$$AXB + X = C \tag{1}$$

where $A \in R^{m\times m}$, $B \in R^{n\times n}$ and $C \in R^{m\times n}$ are the given constant coefficient matrices, while $X \in R^{m\times n}$ is the unknown matrix to be solved. Particularly, if $A = B^T$, the matrix equation (1) can be transformed to the Lyapunov matrix equation [4][5]. In addition, if B in (1) is a nonsingular matrix, post-multiplying (1) by B^{-1} can yield the Sylvester matrix equation [4][6]-[7].

There are two general types of solutions to the problem of algebraic matrix equations. One type of solution is numerical algorithms performed on digital computers. Usually, such numerical algorithms are of serial-processing nature and may not be efficient enough for large-scale online or real-time applications [8]. Being the second type of solution, many parallel-processing methods have

© Springer-Verlag Berlin Heidelberg 2011

been developed and implemented on specific architectures [9]-[13]. Recently, due to the in-depth research on neural networks, numerous dynamic and analog solvers based on recurrent neural networks (RNN) have been developed and investigated [9]-[13]. The neural-dynamic approach is now regarded as a powerful alternative for online computation and optimization owing to its parallel-distributed nature as well as convenience of electronic implementation [11][13].

2 Problem Formulation and Neural Solvers

In this ensuing subsections, ZNN and GNN models are developed and investigated for comparative purposes to solve online the time-varying Stein equation.

2.1 Preliminaries

It follows from [4] that, Ding *et al* have proposed a hierarchical identification principle to solve the constant linear matrix equations with accuracy and effectiveness. In March 2001 [6], Zhang *et al* formally proposed a special kind of RNN for time-varying problems solving, which is different from gradient-based neural networks (GNN) exploited for constant problems solving [9][11][13]. That is, In this work, we mainly consider the following time-varying Stein matrix equation:

$$A(t)X(t)B(t) + X(t) = C(t) \tag{2}$$

where $A(t) \in R^{m\times m}$, $B(t) \in R^{n\times n}$, and $C(t) \in R^{m\times n}$ are smoothly time-varying coefficient-matrices, which, together with their time-derivatives, are assumed to be known numerically or could be estimated accurately. In addition, $X(t) \in R^{m\times n}$ is the time-varying unknown matrix to be solved, and our objective in this work is to find $X(t)$ so that the time-varying Stein equation (2) holds true for any time $t \geqslant 0$. Before solving (2), the following lemmas are presented.

Lemma 1. *[4][5] The time-varying Stein equation (2) is uniquely solvable, if $\lambda_i[A(t)] \cdot \lambda_j[B(t)] \neq -1$ for $\forall i = 1, 2, 3, \cdots, m$ and $j = 1, 2, 3, \cdots, n$ at any time instant $t \in [0, +\infty)$, where $\lambda_i[P(t)]$ denotes the ith eigenvalue of the time-varying matrix $P(t)$.*

Lemma 2. *If Lemma 1 is satisfied, then $M(t) := B^T(t) \otimes A(t) + I$ is a nonsingular time-varying matrix, where I denotes an appropriately-dimensioned identity-matrix, symbol $\otimes$ denotes the Kronecker product [6][10][14], and superscript ^{T} denotes the transpose operator of a matrix or vector.*

2.2 Neural Solver Models Description

To solve online equation (2), we could develop a RNN model by Zhang *et al*'s design method [6,10,12,13]. Firstly, we set up a matrix-valued indefinite error-function $E(t) = A(t)X(t)B(t) + X(t) - C(t) \in R^{m\times n}$, where every element $e_{ij}(t) \in R$ (with $i = 1, 2, 3, \cdots, m$ and $j = 1, 2, 3, \cdots, n$) of error function $E(t)$

could be positive, negative, bounded or even unbounded. Then, to make every element $e_{ij}(t)$ of $E(t)$ converge to zero [i.e., in mathematics, $\lim_{t\to\infty} e_{ij}(t) = 0$], by following ZNN design formula $\dot{E}(t) = -\Gamma\mathcal{F}(E(t))$ [6][10][12][13], we have the following general ZNN model with an implicit-dynamic equation:

$$\begin{aligned} A(t)\dot{X}(t)B(t) + \dot{X}(t) = &- \gamma\mathcal{F}\big(A(t)X(t)B(t) + X(t) - C(t)\big) \\ &- \dot{A}(t)X(t)B(t) - A(t)X(t)\dot{B}(t) + \dot{C}(t) \end{aligned} \tag{3}$$

where the matrix-valued design parameter Γ could simply be γI with constant scalar $\gamma > 0$, $\mathcal{F}(\cdot)$: $R^{m\times n} \to R^{m\times n}$ denotes a matrix activation-function array of neural networks, and $X(t) \in R^{m\times n}$, starting from any initial condition $X(0) := X_0 \in R^{m\times n}$, is the activation state matrix corresponding to the time-varying theoretical solution $X^*(t) \in R^{m\times n}$ of (2). We use $\dot{A}(t) \in R^{m\times m}$, $\dot{B}(t) \in R^{n\times n}$ and $\dot{C}(t) \in R^{m\times n}$ to denote the known numerical forms or measurements of the time-derivatives of matrices $A(t)$, $B(t)$ and $C(t)$, respectively. As for the convergence of ZNN (3), we have the following two propositions [12][13].

Proposition 1. *Consider smoothly time-varying matrices $A(t) \in R^{m\times m}$, $B(t) \in R^{n\times n}$ and $C(t) \in R^{m\times n}$ in (2), which satisfy Lemma 1. If a monotonically-increasing odd activation-function-array $\mathcal{F}(\cdot)$ is used, the neural state $X(t) \in R^{m\times n}$ of ZNN (3), starting from any initial state $X(0) \in R^{m\times n}$, globally converges to the theoretical solution $X^*(t)$ of (2).*

Proposition 2. *In addition to Proposition 1, if the linear function $f(e_{ij}) = e_{ij}$ is employed, then the neural state $X(t)$ of ZNN model (3) could globally converge to the time-varying theoretical solution $X^*(t)$ with the exponential convergence rate γ. Moreover, if we use the power-sigmoid activation function*

$$f(e_{ij}) = \begin{cases} e_{ij}^p, & \text{if } |e_{ij}| \geqslant 1 \\ \frac{1+\exp(-\xi)}{1-\exp(-\xi)} \cdot \frac{1-\exp(-\xi e_{ij})}{1+\exp(-\xi e_{ij})}, & \text{otherwise} \end{cases} \tag{4}$$

with suitable design parameters $p \geqslant 3$ (being an odd integer) and $\xi \geqslant 2$, the neural state $X(t)$ of ZNN (3) is superiorly globally convergent to the theoretical solution $X^*(t)$, as compared to the usage of the linear situation.

For comparison purposes, we can also develop a GNN model to solve (2). According to the conventional GNN, we firstly define a scalar-valued norm-based energy-function, such as $\varepsilon := \|AXB + X - C\|_F^2/2$ with Frobenius norm $\|A\|_F := \sqrt{\text{trace}(A^TA)}$ [10]. Then, evolving along the negative gradient method and applying the time-varying situation, we can obtain a linear GNN model

$$\begin{aligned} \dot{X}(t) = -\gamma\frac{\partial\varepsilon}{\partial X} = &- \gamma\Big(A^T(t)\big(A(t)X(t)B(t) + X(t) - C(t)\big)B^T(t) \\ &+ \big(A(t)X(t)B(t) + X(t) - C(t)\big)\Big), \end{aligned}$$

and also a general nonlinear GNN model

$$\begin{aligned} \dot{X}(t) = &- \gamma\Big(A^T(t)\mathcal{F}\big(A(t)X(t)B(t) + X(t) - C(t)\big)B^T(t) \\ &+ \mathcal{F}\big(A(t)X(t)B(t) + X(t) - C(t)\big)\Big). \end{aligned} \tag{5}$$

3 Computer Simulation Techniques

While Section 2 presents ZNN and GNN models together with related analysis results, in this section, the following MATLAB simulation techniques [15] are employed and investigated to show the characteristics of neural solvers.

3.1 Kronecker Product and Vectorization

Review ZNN (3) and GNN (5). Their dynamic equations are all described in matrix-form, which cannot be directly simulated. Thus, the Kronecker product and vectorization techniques are needed to transform such matrix-form differential equations to vector-form differential equations.

Proposition 3. *Matrix differential equation (3) could be reformulated as the following vector differential equation:*

$$\begin{aligned}(B^T(t)\otimes A(t)+I)\,\mathrm{vec}(\dot{X}(t)) = &-\gamma\mathcal{F}((B^T(t)\otimes A(t)+I)\,\mathrm{vec}(X(t))-\mathrm{vec}(C(t)))\\ &-B^T(t)\otimes\dot{A}(t)\,\mathrm{vec}(X(t))\\ &-\dot{B}^T(t)\otimes A(t)\,\mathrm{vec}(X(t))+\mathrm{vec}(\dot{C}(t))\end{aligned}$$

or simply put,

$$M(t)\dot{Y}(t) = -\gamma\mathcal{F}(M(t)Y(t)-M_3(t)) - M_1(t)Y(t) - M_2(t)Y(t) + \dot{M}_3(t),$$

where $M(t) := B^T(t)\otimes A(t)+I$, $M_1(t) := B^T(t)\otimes\dot{A}(t)$, $M_2(t) := \dot{B}^T(t)\otimes A(t)$, $M_3(t) := \mathrm{vec}(C(t))$, *and* $Y(t) := \mathrm{vec}(X(t))$.

For example, given matrices $A(t)$ and $B(t)$, to generate the mass matrix $M(t) = B^T(t)\otimes A(t)+I$, we have the following user-defined function by exploiting MATLAB routine “kron”.

```
function output=MatrixM(t,x)
A=MatrixA(t,x);[ma,na]=size(A);
B=MatrixB(t,x);[mb,nb]=size(B);
output=kron(B.',A)+eye(ma*mb,na*nb);
```

3.2 Obtaining Matrix Derivatives

In the process of Stein equation solving by using ZNN (3), the time derivatives $\dot{A}(t)$, $\dot{B}(t)$ and $\dot{C}(t)$ are assumed to be known or measurable. This implies that these derivatives can be given directly in an analytical form or can be estimated via finite difference. Thus, to obtain all the time derivatives required in vector-form ZNN model, without loss of generality, we can use MATLAB routine“diff” for such derivatives evaluation. For example, we can obtain the time-derivative of the time-varying matrix $A(t)$ using the following user-defined function.

```
function output=DiffA(t,x)
syms u; A=MatrixA(u);
D=diff(A);
u=t;
output=eval(D);
```

Note that, a symbolic object "u" has to be constructed firstly, and then we could use the command D=diff(A) to generate the analytical form of $\dot{A}(t)$. Finally, evaluating such an analytical form with a numerical value of t will give us the required value of $\dot{A}(t)$. Similarly, we can generate other matrix-derivatives.

3.3 ZNN Right-Hand Side

With the usage of command "reshape", the vectorization of a matrix could be achieved. For example, the following MATLAB code is used to evaluate the right-hand side of vector-form ZNN model.

```
function output=ZnnRightHandSide(t,x,gamma)
A=MatrixA(t,x);B=MatrixB(t,x);C=MatrixC(t,x);
M=MatrixM(t,x);
DotA=DiffA(t,x);DotB=DiffB(t,x);DotC=DiffC(t,x);
[m,n]=size(C);vecC=reshape(C,m*n,1);vecDotC=reshape(DotC,m*n,1);
M1=kron(B.',DotA);
M2=kron(DotB.',A);
err=M*x-vecC;
output=-M1*x-M2*x-gamma*AFMpowersigmoid(err)+vecDotC;
```

In a similar way, we can evaluate the GNN right-hand side.

4 Illustrative Example

In the previous sections, the neural-dynamic models, their theoretical results, and simulation techniques [15] have been presented. For illustration and comparison purposes, ZNN (3) and GNN (5) are both exploited for online solution of the same time-varying Stein matrix equation (2), which are based on the usage of power-sigmoid activation functions with design parameters $p = 3$ and $\xi = 4$.

Consider the Stein matrix equation $A(t)X(t)B(t) + X(t) = C(t)$ with the following time-varying coefficient matrices:

$$A(t) = \begin{bmatrix} 2+\cos 2t & 2\sin 2t \\ \sin 2t & 2-\cos 2t \end{bmatrix}, B(t) = \begin{bmatrix} 2+\cos 2t & \sin 2t & 0 \\ \sin 2t & 1 & \cos 2t \\ \cos 2t & \sin 2t & 2 \end{bmatrix}, \text{ and}$$

$$C(t) = \begin{bmatrix} -2+\cos 2t & 1+\sin 2t & \cos 2t \\ \sin 2t & -2\sin 2t & 1+\cos 2t \end{bmatrix}.$$

Note that, in order to check the correctness of the neural solutions $X(t)$ of ZNN (3) and GNN (5), the time-varying theoretical solution $X^*(t)$ can be obtained by simple algebraic manipulations.

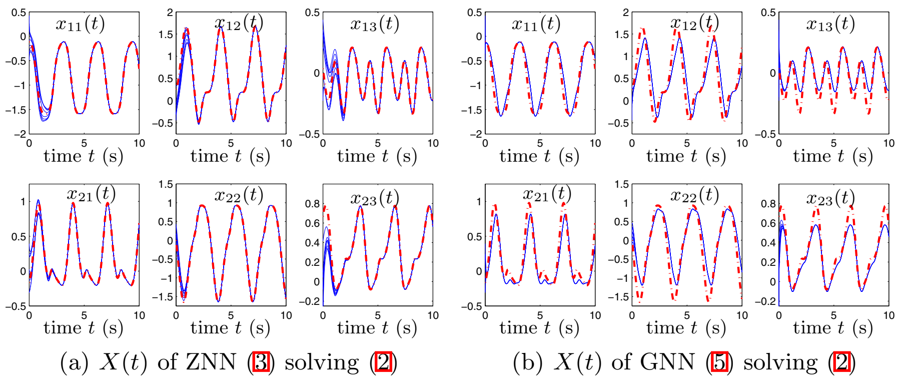


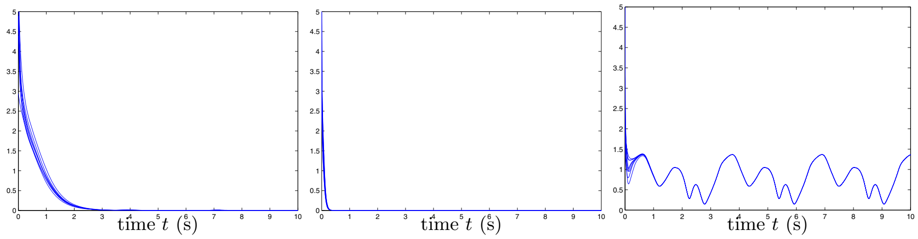
Fig. 1. Online solution of the time-varying linear Stein equation (2) by ZNN and GNN models with $\gamma = 1$, where dash-dotted red curves correspond to the theoretical state $X^*(t)$, and solid blue curves correspond to the neural solution $X(t)$

4.1 Neural-State Simulation

By following ZNN model (3), we could self-define a function used to generate the neural-sate of the Stein matrix equation (2) with the above time-varying coefficients, which could seen in Fig. 1. Part of code is listed as follows.

```
close all;
clear,clc;
gamma=1;
for i=1:8
    x0=(rand(6,1)-0.5*ones(6,1));
    options=odeset('Mass',@MatrixM,'MStateDep','none');
    [t,xz]=ode45(@ZnnRightHandSide,tspan,x0,options,gamma);
    figure (3);subplot(2,3,1);plot(t,xz(:,1));hold on
    figure (3);subplot(2,3,2);plot(t,xz(:,3));hold on
    figure (3);subplot(2,3,3);plot(t,xz(:,5));hold on
    figure (3);subplot(2,3,4);plot(t,xz(:,2));hold on
    figure (3);subplot(2,3,5);plot(t,xz(:,4));hold on
    figure (3);subplot(2,3,6);plot(t,xz(:,6));hold on
end
```

As shown in Fig. 1(a), starting from eight initial states randomly-selected within $[-0.5, 0.5]^{2\times3}$, the neural state $X(t)$ (denoted by solid blue curves) of the presented ZNN model (3) could always converge to the theoretical solution $X^*(t)$ (denoted by dash-dotted red curves) exactly with design parameter $\gamma = 1$. For comparison, GNN model (5) is employed as well to solve (2) under the same conditions similarly. Its convergence could be seen in Fig. 1(b). The GNN-solution could not always fit well with the time-varying theoretical solution $X^*(t)$ with quite large lagging-behind errors. This is evidently because the time-derivative information of the time-varying coefficients $A(t)$, $B(t)$ and $C(t)$ has not been fully utilized in these gradient-based computational schemes.



(a) ZNN (3) with $\gamma = 1$ (b) ZNN (3) with $\gamma = 10$ (c) GNN (5) with $\gamma = 1$

Fig. 2. Residual errors $\|A(t)X(t)B(t) + X(t) - C(t)\|_F$ of ZNN and GNN for (2)

4.2 Residual-Error Simulation

In addition, the residual-error $\|A(t)X(t)B(t)+X(t)-C(t)\|_F$ can also be used to monitor the neural-network convergence characteristics. The user-defined function could be used to generate Fig. 2. Part of code is also attached.

```
close all;
clear,clc;
gamma=1;
for i=1:8
    x0=(rand(6,1)-0.5*ones(6,1));
    options=odeset('Mass',@MatrixM,'MStateDep','none');
    [t,xz]=ode45(@ZnnRightHandSide,tspan,x0,options,gamma);
     for j=1:length(t)
          T=t(j);
          C=MatrixC(T,xz);[m,n]=size(C);vecC=reshape(C,m*n,1);
          M=MatrixM(T,xz);
          errz(:,j)=M*(xz(j,:))'-vecC;
          nerrz(j)=norm(errz(:,j));
     end
     figure(4); plot(t,nerrz);
     hold on
     nerrz=0;
end
```

It can be seen from Fig. 2(a) that, by applying ZNN model (3) to solve online (2), the residual error is convergent to zero within 6 seconds. In contrast, as shown in Fig. 2(c), by using GNN model (5) to solve online equation (2) under the same conditions in a similar way, the residual error is rather large. In addition, it is worth pointing out that, as shown in Figs. 2(a) and (b), the convergence time for ZNN model (3) can be expedited from around 6 seconds to 0.6 second, as the value of design parameter γ is increased from 1 to 10.

5 Conclusions

In this paper, a ZNN model is developed and investigated for the online solution of the time-varying Stein equation $A(t)X(t)B(t) + X(t) = C(t)$ by using

MATLAB simulation techniques. Theoretical analysis demonstrated that the neural state of such a ZNN model can globally exponentially converge to the theoretical solution. In comparison with the conventional neural-network computational approaches, the new design approach and its resultant ZNN models could have much superior efficacy on time-varying problems solving. Computer-simulation results have further substantiated the effectiveness, efficiency and superiority of the presented ZNN models for such time-varying problems solving.

References

1. Zhang, Y., Wang, J.: Global Exponential Stability of Recurrent Neural Networks for Synthesizing Linear Feedback Control Systems via Pole Assignment. IEEE Transactions on Neural Networks 13, 633–644 (2002)
2. Chong, E.K.P., Hui, S., Żak, S.K.: An Analysis of a Class of Neural Networks for Solving Linear Programming Problems. IEEE Transactions on Automatic Control 44, 1995–2006 (1999)
3. Steriti, R.J., Fiddy, M.A.: Regularized Image Reconstruction Using SVD and a Neural Network Method for Matrix Inversion. IEEE Transactions on Signal Processing 41, 3074–3077 (1993)
4. Ding, F., Chen, T.: Gradient based Iterative Algorithms for Solving a Class of Matrix Equations. IEEE Transactions on Automatic Control 50, 1216–1221 (2005)
5. Barraud, A.Y.: Numerical Algorithm to Solve $A^TXA - X = Q$. IEEE Transactions on Automatic Control 22, 883–885 (1977)
6. Zhang, Y., Jiang, D., Wang, J.: A Recurrent Neural Network for Solving Sylvester Equation with Time-Varying Coefficients. IEEE Transactions on Neural Networks 13, 1053–1063 (2002)
7. Wu, A., Duan, G., Zhou, B.: Solution to Generalized Sylvester Matrix Equations. IEEE Transactions on Automatic Control 55, 811–815 (2008)
8. Zhang, Y., Leithead, W.E.: Exploiting Hessian Matrix and Trust-Region Algorithm in Hyperparameters Estimation of Gausssian Process. Applied Mathematics and Computation 171, 1264–1281 (2005)
9. Pearlmutter, B.A.: Gradient Calculations for Dynamic Recurrent Neural Networks: a Survey. IEEE Transactions on Neural Networks 6, 1212–1228 (1995)
10. Zhang, Y., Yi, C., Ma, W.: Simulation and Verification of Zhang Neural Network for Online Time-Varying Matrix Inversion. Simulation Modelling Practice and Theory 17, 1603–1617 (2009)
11. Maeda, Y., Wakamura, M.: Simultaneous Perturbation Learning Rule for Recurrent Neural Networks and its FPGA Implementation. IEEE Transactions on Neural Networks 16, 1664–1672 (2005)
12. Zhang, Y., Ge, S.S.: Design and Analysis of a General Recurrent Neural Network Model for Time-Varying Matrix Inversion. IEEE Transactions on Neural Networks 16, 1477–1490 (2005)
13. Yi, C., Zhang, Y.: Analogue Recurrent Neural Network for Linear Algebraic Equation Solving. Electronics Letters 44, 1078–1079 (2008)
14. Horn, R.A., Johnson, C.R.: Topics in Matrix Analysis, pp. 239–297. Cambridge University Press, Cambridge (1991)
15. Mathews, J.H., Fink, K.D.: Numerical Methods Using MATLAB, 4th edn. Pretice Hall, New Jersey (2004)

Comparison on Continuous-Time Zhang Dynamics and Newton-Raphson Iteration for Online Solution of Nonlinear Equations

Yunong Zhang*, Zhende Ke, Zhan Li, and Dongsheng Guo

School of Information Science and Technology
Sun Yat-sen University, Guangzhou 510006, China
zhynong@mail.sysu.edu.cn

Abstract. Our previous work has shown the efficacy and promising performance of continuous-time Zhang dynamics (CTZD) for solving online nonlinear equations, as compared with conventional gradient dynamics (GD). It has also shown that, with linear activation functions used and with step-size being 1, the discrete-time Zhang dynamics (DTZD) reduces to the Newton-Raphson iteration (NRI) (i.e., being a special case of the DTZD model) for static nonlinear equations solving. It is known that the NRI may fail to converge to the theoretical roots of some difficult or special problems. In this paper, the CTZD model and NRI method are investigated comparatively for online solution of static nonlinear equations by performing numerical tests in different situations. Computer testing and simulation results further demonstrate the efficacy and different convergence-performance of the CTZD model (activated by a power-sigmoid function) and NRI for nonlinear equations solving.

Keywords: Zhang dynamics (ZD), Newton-Raphson iteration (NRI), Numerical comparisons, Nonlinear equations.

1 Introduction

The problem of solving nonlinear equations is considered to be an important issue widely arising in science and engineering fields. Many numerical algorithms (e.g., [1–3] and references therein) have thus been proposed for such a problem solving. However, it may not be efficient enough for many numerical algorithms performed on digital computers in a serial-processing manner [4]. In recent decades, owing to the in-depth research in artificial neural networks, various neural-dynamic (ND) solvers [e.g., recurrent neural networks (RNN)] have been proposed, developed, investigated and implemented [5–10]. Due to the potential suitability for analog VLSI implementation [7, 11] as well as the high-speed processing and parallel-distributed properties, the ND approach is now regarded as a powerful alternative to online problems solving [8–10]. Besides, it is worth

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

mentioning that most computational schemes are theoretically/intrinsically designed for time-invariant problems solving currently, which are usually related to the traditional gradient-based methods [7–10].

Differing from the gradient-based dynamics (GD) approach for static problems solving, a new neural-dynamic solver [termed Zhang neural network (ZNN) or Zhang dynamics (ZD)] has been proposed by Zhang *et al.* [12–14] (formally since March 2001) for time-varying problems solving (e.g., time-varying Sylvester equation solving, time-varying matrix square roots finding, time-varying matrix inversion and optimization). Applied to the aforementioned time-varying situation, the proposed ZD models utilize the time-derivative information of time-varying coefficients of a nonlinear equation, and thus the exact time-varying solution can be obtained in an error-free manner [15, 16]. In contrast, the GD models generate relatively larger errors for the same time-varying problems solving, since no time-derivative information is utilized in GD models [12–16]. By following Zhang *et al*'s design method, the ZD model has recently been introduced to handle static (or termed, time-invariant) nonlinear equations [15, 16].

In our previous work [15–17], theoretical analysis and computer-simulation results about the aforementioned continuous-time ZD (CTZD) model have been given for online solution of both time-varying and static nonlinear equations. It is worth mentioning that, if we utilize a linear activation function and fix the step-size to be 1, the Newton-Raphson iteration (NRI) is obtained as a special case of the discrete-time ZD (DTZD) models (by focusing on static problems solving only) [17, 18]. For NRI, we know that it may fail to converge to some theoretical roots of some special problems [1]. In this paper, the CTZD and NRI are investigated for online solution of static nonlinear equations by performing numerical comparisons in different situations. Computer testing and simulation results demonstrate well the efficacy and different advantages of CTZD (activated by power-sigmoid functions) and NRI for solving static nonlinear equations.

2 Problem Formulation and Solvers

Our main purpose in this paper is to solve the nonlinear equation depicted by

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

where the unknown scalar $x \in R$ is to be obtained, and nonlinear mapping $f(\cdot) : R \rightarrow R$ is assumed a continuously-differentiable function [15–17]. For ease of presentation, let x^* denote a theoretical solution (or termed, root, zero) of nonlinear equation (1). In the ensuing subsections, both ZD model and NRI method are generalized, developed and exploited to solve (1) comparatively.

2.1 CTZD Model

To monitor and control the solution process of nonlinear equation (1), by following Zhang *et al*'s neural-dynamic design method [12–15], we firstly define the following Zhang function $E(t)$, being a form of error-monitoring function, which is evidently indefinite and lower-unbounded:

$$E(t) := f\big(x(t)\big).$$

Secondly, the time derivative $\dot{E}(t)$ of $E(t)$ should be chosen such that $E(t)$ (globally) exponentially converges to zero. Specifically, $\dot{E}(t)$ is chosen via the following Zhang *et al*'s design formula [15–17]:

$$\frac{dE(t)}{dt} = -\gamma\phi\big(E(t)\big), \quad \text{or} \quad \frac{df(x)}{dt} = -\gamma\phi\big(f(x)\big). \tag{2}$$

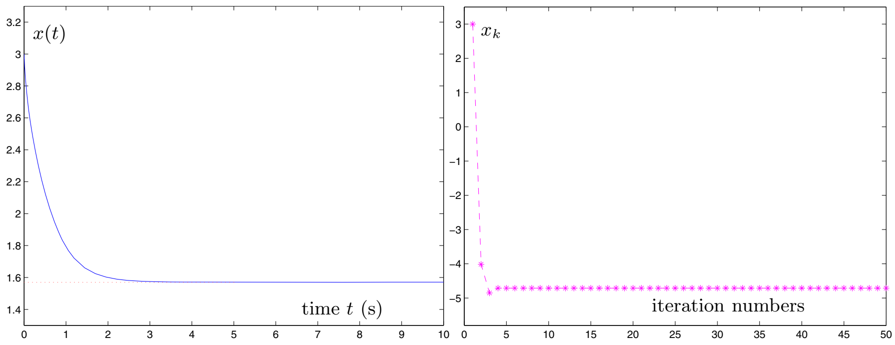
In equation (2), the design-parameter $\gamma > 0$, being used to control the convergence rate of the CTZD model, should be set suitably large for simulation purposes. The activation function $\phi(\cdot) : R \to R$ is assumed to be monotonically-increasing and odd. Expanding the above ZD design formula (2), we obtain a CTZD model depicted in the following explicit dynamics if $f'(x) \neq 0$ [with time $t \in [0, +\infty)$, $f'(x) := \partial f(x)/\partial x$ and $\dot{x} := dx/dt$]:

$$\dot{x} = -\gamma\frac{\phi\big(f(x)\big)}{f'(x)}, \tag{3}$$

where $x(t)$ is the state as well as the output of CTZD model (3) corresponding to theoretical root x^* of nonlinear equation (1). Note that different performance of CTZD model (3) is achieved by using different design-parameter γ and activation function $\phi(\cdot)$ [15–17]. In general, any monotonically-increasing odd function can be the activation function of the CTZD model. Since March 2001 [12], we have introduced and used five types of activation functions (i.e., linear activation function, power activation function, power-sum activation function, sigmoid activation function and power-sigmoid activation function) for the ZD models (for more details, see [10, 13, 15, 19]). Besides, it is worth pointing out that the pitfall of the CTZD model is the possibility of division by zero in equation (3), which would occur if $f'(x) = 0$; and that, for nonlinear equation (1) involving points with $f'(x) = 0$, the initial condition $x(0)$ has to be chosen close enough to x^* and thus root x^* is obtained.

For CTZD model (3) solving the nonlinear equation (1) with no local minima, we have the following proposition about its convergence [15, 16].

Proposition 1. *Consider a solvable nonlinear equation $f(x) = 0$, where function $f(\cdot)$ is continuously differentiable. If a monotonically-increasing odd activation function $\phi(\cdot)$ is used, then the state $x(t)$ of CTZD model (3), starting from randomly-generated initial condition $x(0) := x_0 \in R$, can converge to theoretical root x^* of nonlinear equation $f(x) = 0$. Note that the specific value of x^*, in the situation of not less than two zeros existing, depends on the sufficient closeness of initial state x_0.*



(a) Right state trajectory of CTZD (3) (b) Wrong state trajectory of NRI (4)

Fig. 1. Online solution of nonlinear equation (5) by CTZD model (3) with $\gamma = 1$ and NRI method (4), starting from the same initial state $x(0) = x_0 = 3$

Review CTZD model (3) further. If we utilize a linear activation function $\phi(x) = x$ and fix the design parameter γ to be 1, the CTZD model (3) reduces to the following so-called continuous Newton's method interestingly [20]:

$$\dot{x} = -f(x)/f'(x).$$

2.2 NRI Method

Differing from the traditional (or to say, standard) explanation to NRI method appearing in almost all literature and textbooks, i.e., via Taylor series expansion [1, 7], we have discovered once more that a general form of NRI is obtained by discretizing the CTZD model (3). Specifically, the NRI for solving nonlinear equations can be derived by focusing on the static problem solving (1), discretizing CTZD model (3) via Euler forward difference, using a linear activation function and fixing the step-size to be 1 [17, 18]. Thus, we obtain the NRI for online solution of nonlinear equation (1) as follows:

$$x_{k+1} = g(x_k) := x_k - \frac{f(x_k)}{f'(x_k)},\ k = 0, 1, 2, 3, \cdots . \tag{4}$$

It is known that x_k in NRI (4) may fail to converge to a theoretical root when an initial condition is improperly chosen [1]. So far, many convergence results about NRI have been introduced and presented (e.g., [1, 20, 21] and references therein), and most of them are on the choices of initial condition and the estimation of convergence speed (i.e., quadratic for simple roots and linear for multiple roots).

Remarks. *It is worth comparing the two methods of CTZD (3) and NRI (4), both of which are now exploited for online solution of static nonlinear equation*

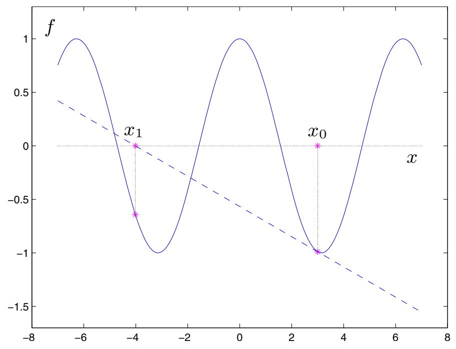


Fig. 2. The graphical interpretation of the failure of NRI (4) when solving (5)

(1). In addition to the main difference being ZD for time-varying problems solving and NRI for static problems solving, other differences lie in the following.

1) CTZD (3) is a continuous-time method, of which the design method is based on the elimination of an indefinite error-function $E(t) := f\big(x(t)\big)$ which can be positive, negative, bounded or even unbounded. In contrast, NRI (4) is a discrete-time (iterative) method, which is traditionally thought to be obtained from Taylor series expansion [or newly thought in our research to be derived by discretizing CTZD (3) with some specific conditions].

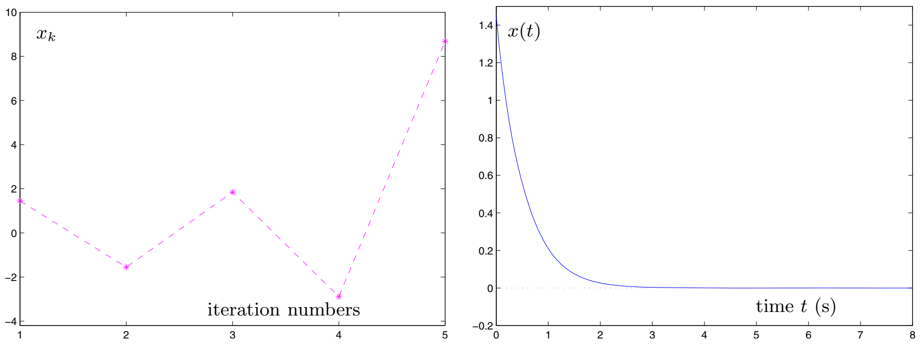
2) The convergence behavior of CTZD (3) is that the state $x(t)$, starting from an initial condition, traverse every point of the curve until the root is found. In contrast, the convergence behavior of NRI (4) is that the current point jumps to the next point on the function curve by finding the intersection between the x axis and the line tangent to the curve of $y = f(x)$ at the current point.

3) The theoretical analysis on convergence of CTZD (3) is based on the well-known Lyapunov theory or ordinary differential equations [15]. Differing from that of CTZD, the standard theoretical analysis on convergence of NRI (4) is based on the well-known fixed-point theorem and related knowledge [1].

4) The convergence speed of CTZD (3) can be expedited by increasing γ and using a suitable activation function. In contrast, the convergence rate for NRI (4) is fixed, which is quadratic for simple roots and linear for multiple roots [1].

3 Computer Testing and Simulation Results

The previous sections have addressed the CTZD model (3) and the NRI method (4) for solving nonlinear equations. In this section, by performing numerical comparisons in different situations, computer testing and simulation results are provided to show the characteristics of the two solvers. Though NRI (4) can be obtained from CTZD (3), their convergence behaviors are quite different from



(a) Divergent trajectory of NRI (4) (b) Convergent trajectory of CTZD (3)

Fig. 3. Online solution of (6) by NRI (4) and CTZD (3) with $\gamma = 1$, starting from the same initial state $x_0 = x(0) = 1.45$

each other, which evidently results in different advantages of CTZD (3) and NRI (4) for solving nonlinear equations. To investigate and compare the efficacy of such two solvers, let us consider the following illustrative examples.

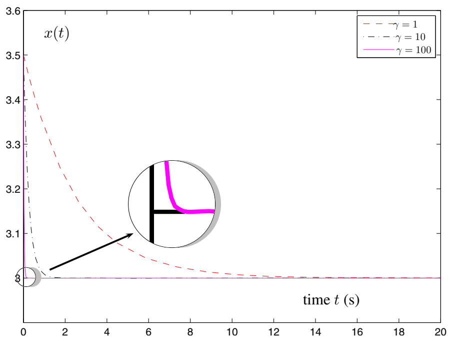
Example 1. $f(x)$ **with** x **defined in a range** $[a, b]$

It is widely encountered in science and engineering problems that the variable x in nonlinear function $f(x)$ is defined only in a finite range, e.g., $[a, b]$. For illustration and comparison, let us consider the following nonlinear equation:

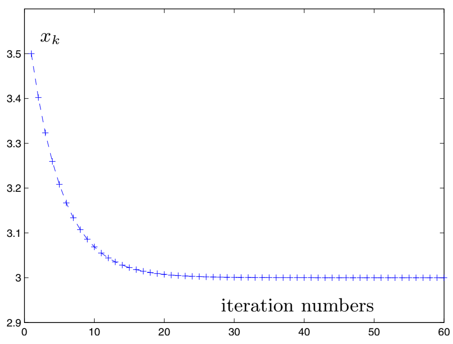
$$f(x) = \cos(x) = 0, \text{ with } x \in [0, \pi]. \tag{5}$$

Evidently, the only theoretical root in the defining range $[0, \pi]$ is $x^* = \pi/2$. Both CTZD (3) and NRI (4) are then exploited to solve the above nonlinear equation (5). By using CTZD (3) with $\gamma = 1$, we see from Fig. 1(a) that, starting from initial state $x(0) = 3$, the state $x(t)$ converges to the desired theoretical root $x^* = \pi/2 \approx 1.57$ after a short time (e.g., 3 s or so). But, for the same initial state $x_0 = 3$, as shown in Fig. 1(b), the state x_k of NRI (4) converges to a different root $-3\pi/2 \approx -4.71$ which is out of range $[0, \pi]$. Actually, the state calculated from the first iteration $x_1 = x_0 - f(x_0)/f'(x_0) = -4.015$ has already been out of range $[0, \pi]$. Simply put, CTZD (3) generates a right solution, whereas NRI (4) generates a wrong solution.

As seen from the simulation results, CTZD (3) has superior effectiveness to NRI (4) in the situation of x defined in a range $[a, b]$ according to their convergence behaviors discussed in the aforementioned remarks. In addition, the initial condition of NRI (4) should be chosen close enough to the desired theoretical root; otherwise, it may converge to some other root. Moreover, as seen from Fig. 2, the absolute value of the slope $f'(x_0)$ should not be small; otherwise [e.g., the slope $f'(x_0)$ of (5) is about -0.14], the tangent line of function $f(x)$ at point x_0 is nearly horizontal and the first intersection x_1 is far away from x_0, leading to the consequence $\{x_k\}$ running out of the defining range $[0, \pi]$.



(a) Expedited trajectories of CTZD (3)



(b) Fixed state trajectory of NRI (4)

Fig. 4. Online solution of (7) by CTZD model (3) with $\gamma = 1$, 10 and 100 as well as NRI method (4), starting from the same initial state $x(0) = x_0 = 3.5$

Example 2. The divergent oscillating case of NRI (4)

According to the fixed-point theorem given in [1], NRI (4) may oscillate and diverge when $|g'(x)| \geq 1$ on an interval containing root x^*. For illustration and comparison, let us consider the following nonlinear equation:

$$f(x) = \arctan(x) = 0. \tag{6}$$

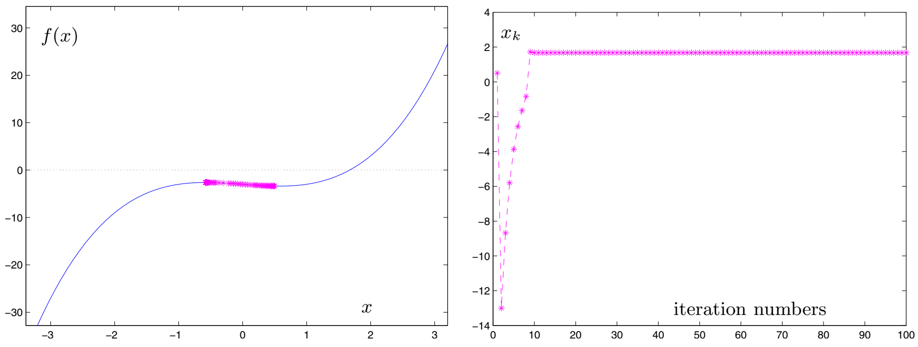
Evidently, one theoretical root is $x^* = 0$. By using NRI (4) to solve nonlinear equation (6), we obtain $g'(x) = -2x \cdot \arctan(x)$ [1]. As seen from Fig. 3(a), the initial state is set as $x_0 = 1.45$, and then the sequence $\{x_k\}$ generated by NRI (4) is divergently oscillating (with $x_1 = -1.55, x_2 = 1.85, x_3 = -2.89, \cdots$). In contrast, as shown in Fig. 3(b), the state $x(t)$ of CTZD (3), starting from the same initial state, converges to theoretical root $x^* = 0$ after a short time. All of these demonstrate the superior effectiveness of CTZD (3) [as compared to NRI (4)], in addition to the important link between them.

Example 3. Convergence speed

To compare the convergence speed of such two models [i.e., CTZD (3) and NRI (4)], let us consider the following nonlinear equation:

$$f(x) = (x + 1)(x - 3)^5 = 0. \tag{7}$$

Evidently, this nonlinear equation (7) theoretically has a simple root $x_1^* = -1$ and a multiple root $x_2^* = 3$ of order five. As shown in Fig. 4(a), starting from initial state $x(0) = 3.5$, the state $x(t)$ of CTZD (3) with $\gamma = 10$ converges to the multiple root $x_2^* = 3$ (about 1.4 s) much faster than the state with $\gamma = 1$ (about 14 s). Moreover, if $\gamma = 100$, the convergence time is 0.14 s. Thus, CTZD (3) has an exponential-convergence property, and the convergence speed can be expedited effectively by increasing the value of design parameter γ. In contrast, for the same initial state, as seen from Fig. 4(b), the convergence rate of NRI



(a) Abortive $x(t)$ movement of CTZD (3) (b) Successful trajectory of NRI (4)

Fig. 5. Convergence performance of CTZD (3) with $\gamma = 1$ and NRI (4) for online solution of nonlinear equation (8), starting from the same initial state $x(0) = x_0 = 0.5$

(4) is fixed, which takes about 25 iterations. Furthermore, for root $x^* = 3$, the convergence rate for NRI (4) is linear as it is a multiple root [1].

Example 4. Nonlinear equation containing a local minimum

Let us consider the situation of nonlinear equation containing a local minimum. According to the different convergence behaviors discussed in the aforementioned remarks, the state $x(t)$ of CTZD (3) may move towards the local minimum point and then stop; in contrast, the state x_k of NRI (4) may converge to the theoretical root after a few iterations. For further illustration and comparison, let us consider the following nonlinear equation:

$$f(x) = x^3 - x - 3 = 0. \tag{8}$$

The theoretical root we consider is $x^* = 1.6717$. As shown in Fig. 5(a) and (b), starting from initial state $x(0) = 0.5$, the state $x(t)$ of CTZD (3) with $\gamma = 1$ can not converge to the theoretical root x^*; instead, it stops at the point $x = -0.5774$ which is the local minimum of nonlinear equation (8). In contrast, starting from the same initial state, the state x_k of NRI (4) converges to the theoretical root after a few iterations. These demonstrate that the NRI method (4) could have a superior effectiveness when nonlinear function $f(x)$ possesses a local minimum, as compared to the CTZD model (3). Note that, as observed from the related simulation results, when we choose the initial state $x_0 = 0.6$ or a bigger value, the state $x(t)$ of CTZD (3) with $\gamma = 1$ converges to the theoretical root. Thus, if nonlinear function $f(x)$ possesses a local minimum, the initial state of CTZD (3) has to be set close enough to the theoretical root so as for $x(t)$ to avoid falling into the basin of attraction of the local minimum of nonlinear function $f(x)$.

4 Conclusions

A special class of neural dynamics (i.e., Zhang dynamics, ZD) has been discovered, developed and analyzed for online solution of time-varying problems. By following and generalizing Zhang *et al*'s design method, the ZD model has recently been introduced to handle the static (or termed, time-invariant) nonlinear equations. In this paper, as an extension of our previous work, comparisons on CTZD and NRI have been made and illustrated for online solution of static nonlinear equations. Both theoretical analysis and simulation results have shown the efficacy and different superiorities of CTZD ([3]) and NRI ([4]) for solving nonlinear equations, according to their different convergence behaviors.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grants 61075121, 60935001 and 60775050, and also by the Fundamental Research Funds for the Central Universities of China. In addition, Yunong would like to thank and encourage the student coauthors with the following sentence "We, researchers, are also travelers: the published papers are just the footprints we made or the sights we saw". Thanks and best regards.

References

1. Mathews, J.H., Fink, K.D.: Nemerical Methods Using MATLAB. Publishing House of Electronics Industry, Beijing (2005)
2. Sharma, J.R.: A Composite Third Order Newton-Steffensen Method for Solving Nonlinear Equations. Applied Mathematics and Computation 169, 242–246 (2005)
3. Chun, C.: Construction of Newton-Like Iteration Methods for Solving Nonlinear Equations. Numerical Mathematics 104, 297–315 (2006)
4. Zhang, Y., Leithead, W.E., Leith, D.J.: Time-series Gaussian Process Regression Based on Toeplitz Computation of O(N^2) operations and O(N)-Level Storage. In: Proceedings of the 44th IEEE Conference on Decision and Control, Seville, Spain, pp. 3711–3716 (2005)
5. Hirsch, M.W., Smale, S.: Differential Equations, Dynamic Systems and Linear Algebra. Academic Press, New York (1974)
6. Zhang, Y.: Dual Neural Networks: Design, Analysis, and Application to Redundant Robotics. In: Progress in Neurocomputing Research, pp. 41–81. Nova Science Publishers, New York (2007)
7. Mead, C.: Analog VLSI and Neural Systems. Addison-Wesley, Reading (1989)
8. Zhang, Y., Wang, J.: Global Exponential Stability of Recurrent Neural Networks for Synthesizing Linear Feedback Control Systems via Pole Assignment. IEEE Transactions on Neural Network 13, 633–644 (2002)
9. Zhang, Y., Wang, J.: A Dual Neural Network for Convex Quadratic Programming Subject to Linear Equality and Inequality Constraints. Physics Letters A 298, 271–278 (2002)
10. Zhang, Y.: Revisit the Analog Computer and Gradient-Based Neural System for Matrix Inversion. In: Proceedings of IEEE International Symposium on Intelligent Control, Limassol, Cyprus, pp. 1411–1416 (2005)

11. Zhang, Y., Ma, W., Li, K., Yi, C.: Brief History and Prospect of Coprocessors. China Science and Technology Information 13, 115–117 (2008)
12. Zhang, Y., Jiang, D., Wang, J.: A Recurrent Neural Network for Solving Sylvester Equation with Time-Varying Coefficients. IEEE Transactions on Neural Network 13, 1053–1063 (2002)
13. Zhang, Y., Ge, S.S.: Design and Analysis of a General Recurrent Neural Network Model for Time-Varying Matrix Inversion. IEEE Transactions on Neural Network 16, 1477–1490 (2005)
14. Zhang, Y., Li, Z.: Zhang Neural Network for Online Solution of Time-Varying Convex Quadratic Program Subject to Time-Varying Linear-Equality Constraints. Physics Letters A 373, 1639–1643 (2009)
15. Zhang, Y., Yi, C., Ma, W.: Comparison on Gradient-Based Neural Dynamics and Zhang Neural Dynamics for Online Solution of Nonlinear Equations. In: Proceedings of International Symposium on Intelligence Computation and Applications, Wuhan, China, pp. 269–279 (2008)
16. Zhang, Y., Xu, P., Tan, N.: Further Studies on Zhang Neural-Dynamics and Gradient Dynamics for Online Nonlinear Equations Solving. In: Proceedings of the IEEE International Conference on Automation and Logistics, Shenyang, China, pp. 566–571 (2009)
17. Zhang, Y., Xu, P., Tan, N.: Solution of Nonlinear Equations by Continuous and Discrete-Time Zhang Dynamics and More Importantly Their Links to Newton Iteration. In: Proceedings of the IEEE International Conference on Information, Communications and Signal Processing, Macau, China, pp. 1–5 (2009)
18. Zhang, Y., Ke, Z., Xu, P., Yi, C.: Time-Varying Square Roots Finding via Zhang Dynamics versus Gradient Dynamics and the Former's Link and New Explanation to Newton-Raphson Iteration. Information Processing Letters 110, 1103–1109 (2010)
19. Li, Z., Zhang, Y.: Improved Zhang Neural Network Model and its Solution of Time-Varying Generalized Linear Matrix Equations. Expert Systems with Applications 37, 7213–7218 (2010)
20. Saupe, D.: Discrete versus Continuous Newton's Method: A Case Study. Acta Applicandae Mathematicae 13, 59–80 (1988)
21. Galántai, A.: The Theory of Newton's Method. Journal of Computational and Applied Mathematics 124, 25–44 (2000)

ELM-Based Time-Variant Neural Networks with Incremental Number of Output Basis Functions

Yibin Ye, Stefano Squartini, and Francesco Piazza

A3LAB, Department of Biomedics, Electronics and Telecommunications,
Università Politecnica delle Marche, Via Brecce Bianche 1, 60131 Ancona, Italy
{yibin.ye,s.squartini,f.piazza}@univpm.it
http://www.a3lab.dibet.univpm.it

Abstract. An Extreme Learning Machine (ELM) approach has already been applied to Time-Variant Neural Networks (TV-NN) with greatly reduced training time. However, several parameters need to be tuned in ELM-TV-NN, such as number of hidden neurons, number of basis functions. Interesting approaches have been proposed to automatically determine the number of hidden nodes in our previous work. In this paper, we explored a way to extend the Error Minimized Extreme Learning Machine (EM-ELM) algorithm along with one incremental based ELM method to the output basis functions case study. Simulation results show the effectiveness of the approach.

1 Introduction

A fast learning algorithm called Extreme Learning Machine (ELM) introduced by Huang et al[1,2], has recently caught much attention within the Neural Network (NN) research community. ELM is designed for single hidden layer feedforward neural networks (SLFNs): it randomly chooses hidden nodes and then determines the output weight analytically. However, one of the open problem of ELM is how to assign the number of hidden neurons, which is the only factor that needs to be set by users, usually by trial-and-error. The first hidden nodes increment algorithm for ELM, referred to as Incremental Extreme Learning Machine (I-ELM)[3], randomly adds nodes to the hidden layer one by one and freezes the output weights of the existing hidden nodes when a new hidden node is added. Another ELM-based hidden nodes incremental learning algorithm referred to as Error Minimized Extreme Learning Machine (EM-ELM)[4] can also add random hidden nodes to SLFNs one by one or even group by group, with all the previous output weights updated accordingly at each step. Compared with I-ELM which keeps the output weights of existing hidden nodes fixed when adding a new hidden node, EM-ELM attain a much better generalization performance.

Time-Variant Neural Networks(TV-NN) represent a relevant example in the field of neural architectures working properly in non-stationary environments. Such networks have time-variant weights, each being a linear combination of a certain set of basis functions. Titti et al. [5], proposed an extended version

© Springer-Verlag Berlin Heidelberg 2011

of the Back-Propagation(BP) algorithm to suitably train such networks. An Extreme Learning Machine approach is also developed for TV-NN(ELM-TV-NN), introduced by Cingolani et al[6], accelerating the training procedure significantly. However, ELM-TV-NN also has the problem to determine the size of network, e.g. the number of hidden nodes and the number/type of basis functions.

Recently, some interesting variants have been studied on purpose. One [7] is oriented to apply a Group-selection theory approach to optimally size the network and choose the type of basis functions, whereas another one proposes some incremental-based learning algorithms for ELM-TV-NN [8], able to automatically determine the number of hidden nodes, avoiding to use the trial-and-error method typically adopted in standard ELM-TV.

In this paper, we attempt to extend I-ELM and EM-ELM to output basis functions of time-variant networks, as well as testing and comparing them in two different task scenarios. The ELM approach for time-variant neural networks is briefly introduced in Section 2, followed by our proposal algorithms presented in Section 3, and the simulation results in Section 4. Finally, conclusion is given in Section 5. Due to the space constraint, we do not present more details for ELM algorithm and its related incremental-based variants I-ELM and EM-ELM. Please refer to [2,3,4] for more information.

2 The ELM Approach for Time-Variant Neural Networks

The application and extension of ELM to the time-variant case has been studied in [6]. In a time-variant neural network, the input weights, or output weights, or both are changing through the training and testing time. Each weight can be expressed as a linear combination of a certain set of basis functions: $w[n] = \sum_{b=1}^{B} f_b[n] \cdot w_b$, in which $f_b[n]$ is the known orthogonal function at n-th time instant of b-th order, w_b is the b-th order coefficient of the basic function to construct time-variant weight w_n, while B is the total number of the bases preset by user.

If time-variant input weights are introduced in a SLFN, hidden neuron output function can be rewritten as: $h_k[n] = g\left(\sum_{i=0}^{I} x_i[n] \cdot w_{ik}[n]\right)$, where $w_{ik}[n] = \sum_{b=1}^{B} f_b[n] \cdot w_{b,ik}$. Similarly, if time-variant output weights are introduced, the standard output equation can be rewritten as: $\sum_{k=1}^{K} h_k[n] \cdot \beta_{kl}[n] = t_l[n]$, where $\beta_{kl}[n] = \sum_{b=1}^{B} f_b[n] \cdot \beta_{b,kl}$.

To train a TV-NN, $\{w_{b,ik}\}$ can be randomly generated and then hidden-layer output matrix $\mathbf{H}$ can be computed. However, the values of a set of output weight parameters $\{\beta_{b,kl}\}$ can not be so straightforward calculated. Some transformations are needed. Let us expand the time-variant output weights $\beta_{kl}[n]$ and assume the following notation: $\mathbf{f}[n] = [f_1[n], f_2[n], \ldots, f_B[n]]^T \in \mathbb{R}^B$, $\mathbf{h}[n] = [h_1[n], h_2[n], \ldots, h_K[n]]^T \in \mathbb{R}^K$, $\boldsymbol{\beta}_{b,l} = [\beta_{b,1l}, \beta_{b,2l}, \ldots, \beta_{b,Kl}]^T \in \mathbb{R}^K$,

$\boldsymbol{\beta}_{(l)} = [\boldsymbol{\beta}_{1,l}, \boldsymbol{\beta}_{2,l}, \ldots, \boldsymbol{\beta}_{B,l}] \in \mathbb{R}^{K \times B}$, $\boldsymbol{\omega}_{(l)} = [\boldsymbol{\beta}_{1,l}^T, \boldsymbol{\beta}_{2,l}^T, \ldots, \boldsymbol{\beta}_{B,l}^T]^T \in \mathbb{R}^{K \cdot B \times 1}$. We can now state:

$$\begin{aligned} t_l[n] &= \sum_{k=1}^{K} h_k[n] \cdot \left(\sum_{b=1}^{B} f_b[n] \cdot \beta_{b,kl} \right) \\ &= \mathbf{h}[n]^T \cdot \boldsymbol{\beta}_{(l)} \cdot \mathbf{f}[n] \\ &= \left(\mathbf{f}[n]^T \otimes \mathbf{h}[n]^T \right) \cdot \boldsymbol{\omega}_{(l)} \end{aligned} \tag{1}$$

where $\otimes$ denotes the *Kronecker product* of $\mathbf{f}[n]^T$ and $\mathbf{h}[n]^T$. The last step consists in: $vec(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A})vec(\mathbf{B})$, (note that $vec(t_l[n]) = t_l[n]$.) and $\boldsymbol{\omega}_{(l)} = vec(\boldsymbol{\beta}_{(l)})$ is the vectorization of the matrix $\boldsymbol{\beta}_{(l)}$ formed by stacking the columns of $\boldsymbol{\beta}_{(l)}$ into a single column vector.

Moreover, we define: $\mathbf{G} = \mathbf{F} * \mathbf{H}$, where $\mathbf{F} = [\mathbf{f}[1], \mathbf{f}[2], \ldots, \mathbf{f}[N]]^T = \{f_b[n]\} \in \mathbb{R}^{N \times B}$, $\mathbf{H} = [\mathbf{h}[1], \mathbf{h}[2], \ldots, \mathbf{h}[N]]^T = \{h_k[n]\} \in \mathbb{R}^{N \times K}$, $*$ denotes the *Khatri-Rao product* of matrices $\mathbf{F}$ and $\mathbf{H}$, with $\mathbf{f}[n]^T$ and $\mathbf{h}[n]^T$ as their submatrices, respectively. Further assuming that: $\mathbf{T} = \{t_l[n]\} \in \mathbb{R}^{N \times L}$, $\boldsymbol{\Omega} = [\boldsymbol{\omega}_{(1)}, \boldsymbol{\omega}_{(2)}, \ldots, \boldsymbol{\omega}_{(L)}] \in \mathbb{R}^{K \cdot B \times L}$, we get: $\mathbf{G} \cdot \boldsymbol{\Omega} = \mathbf{T}$

Since $\mathbf{F}$ is obtained by the type of the basis function predetermined by the user; $\mathbf{H}$ can be calculated once input weight parameters are randomly generated. Hence we can get $\mathbf{G}$. Similar to the ELM algorithm described in previous section, the time-variant output weight matrix $\boldsymbol{\Omega}$ can be computed by:

$$\hat{\boldsymbol{\Omega}} = \mathbf{G}^{\dagger} \cdot \mathbf{T} \tag{2}$$

where $\mathbf{G}^{\dagger}$ is the MP inverse of matrix $\mathbf{G}$, and consequently, $\hat{\boldsymbol{\Omega}}$ is a set of optimal output weight parameters minimizing the training error.

3 Proposed Output Basis Functions ELM Algorithms for TV-NN

3.1 Incremental Output Basis Functions for ELM-TV

It is proved in [3] and [9] that for one specific output neuron, when the k-th hidden neuron is added and $\beta_k = \frac{\tilde{\mathbf{h}}_k^T \cdot \tilde{\mathbf{e}}_{k-1}}{\tilde{\mathbf{h}}_k^T \cdot \tilde{\mathbf{h}}_k}$, $\|\tilde{\mathbf{e}}_k\| = \|\tilde{\mathbf{t}} - (\tilde{\mathbf{t}}_{k-1} + \beta_k \tilde{\mathbf{h}}_k)\|$ achieves its minimum and the sequence $\{\|\tilde{\mathbf{e}}_k\|\}$ decreases and converges. Note that $\beta_k = \frac{\tilde{\mathbf{h}}_k^T \cdot \tilde{\mathbf{e}}_{k-1}}{\tilde{\mathbf{h}}_k^T \cdot \tilde{\mathbf{h}}_k}$, is a special case of $\beta_k = \tilde{\mathbf{h}}_k^{\dagger} \tilde{\mathbf{e}}_{k-1}$ when $\tilde{\mathbf{e}}_{k-1}$ and $\tilde{\mathbf{h}}_k$ are vectors. Actually, the MP generalized inverse of $\tilde{\mathbf{h}}_k$ is just $\tilde{\mathbf{h}}_k^{\dagger} = (\tilde{\mathbf{h}}_k^T \cdot \tilde{\mathbf{h}}_k)^{-1} \tilde{\mathbf{h}}_k^T$ For our time-variant output basis functions case, this can also be extended to matrix computations. Let $\delta\boldsymbol{\Omega}_b = \begin{bmatrix} \beta_{b,11}, \cdots, \beta_{b,1L} \\ \vdots, \ddots, \vdots \\ \beta_{b,K1}, \cdots, \beta_{b,KL} \end{bmatrix}_{K \times L}$ and $\delta\mathbf{G}_b = \begin{bmatrix} f_b[1] \cdot \mathbf{h}[1]^T \\ \vdots \\ f_b[N] \cdot \mathbf{h}[N]^T \end{bmatrix}_{N \times K}$,

(Note $\boldsymbol{\Omega} = \begin{bmatrix} \delta\boldsymbol{\Omega}_1 \\ \vdots \\ \delta\boldsymbol{\Omega}_B \end{bmatrix}$ and $\mathbf{G} = [\delta\mathbf{G}_1, \cdots, \delta\mathbf{G}_B]$.) Similarly, if $\delta\boldsymbol{\Omega}_b = \delta\mathbf{G}_b^{\dagger} \cdot \mathbf{E}_{b-1}$,

$\|\mathbf{E}_b\| = \|\mathbf{T} - (\mathbf{T}_{b-1} + \delta\mathbf{G}_b\delta\mathbf{\Omega}_b)\|$ achieve its minimum and the sequence $\{\|\mathbf{E}_b\|\}$ would decrease and converges.

Hence, we have our time-variant output basis functions version of I-ELM. Assume we have a set of training data $\{(\mathbf{x}[n], \mathbf{t}[n])\}_{n=1}^{N}$, the target output matrix $\mathbf{T}$, the residual matrix $\mathbf{E}$, the maximum number of output basis functions B_{max}, and the expected learning accuracy ϵ. We get Algorithm 1.

Algorithm 1. I-OB

```
1: Generate random input weights $\{w_{b,ik}\}$ and calculate $\mathbf{H}$.
2: Let $b = 0$ and residual error $\mathbf{E} = \mathbf{T}$,
3: while $b < B_{max}$ and $\|\mathbf{E}\| > \epsilon$, do
4:    Increase by one the number of output basis functions: $b = b + 1$;
5:    Calculate the hidden layer output submatrix for new output basis
          $$\delta\mathbf{G}_b = \begin{bmatrix} f_b[1] \cdot \mathbf{h}[1]^T \\ \vdots \\ f_b[N] \cdot \mathbf{h}[N]^T \end{bmatrix}_{N \times K}$$
6:    Calculate the output weight $\delta\mathbf{\Omega}_b$ for the new output basis
          $$\delta\mathbf{\Omega}_b = \delta\mathbf{G}_b^{\dagger} \cdot \mathbf{E}$$
7:    Calculate the residual error after adding the new output basis $b$:
          $$\mathbf{E} = \mathbf{E} - \delta\mathbf{G}_b \cdot \delta\mathbf{\Omega}_b$$
8: end while
9: The output weight matrix would be $\mathbf{\Omega} = \begin{bmatrix} \delta\mathbf{\Omega}_1 \\ \vdots \\ \delta\mathbf{\Omega}_B \end{bmatrix}_{K \cdot B \times L}$
```

3.2 Error Minimized Output Basis Functions for ELM-TV

Similarly, with proper modifications, the Error Minimized ELM approach can also be extended to the time-variant output basis functions case study. Replace $\mathbf{H}_0$, $\delta\mathbf{H}_k$, β with $\mathbf{G}_1$,$\delta\mathbf{G}_b$, $\mathbf{\Omega}$, respectively. With $\mathbf{G}_{b+1} = [\mathbf{G}_b, \delta\mathbf{G}_b]$, we have

$$\mathbf{G}_{b+1}^{\dagger} = (\mathbf{G}_{b+1}^T\mathbf{G}_{b+1})^{-1}\mathbf{G}_{b+1} = \begin{bmatrix} \mathbf{G}_b^T\mathbf{G}_b, & \mathbf{G}_b^T\delta\mathbf{G}_b \\ \delta\mathbf{G}_b^T\mathbf{G}_b, & \delta\mathbf{G}_b^T\delta\mathbf{G}_b \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{G}_b^T \\ \delta\mathbf{G}_b^T \end{bmatrix} \quad (3)$$

Denote

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11}, & \mathbf{A}_{12} \\ \mathbf{A}_{21}, & \mathbf{A}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{G}_b^T\mathbf{G}_b, & \mathbf{G}_b^T\delta\mathbf{G}_b \\ \delta\mathbf{G}_b^T\mathbf{G}_b, & \delta\mathbf{G}_b^T\delta\mathbf{G}_b \end{bmatrix}^{-1} \quad (4)$$

and

$$\mathbf{G}_{b+1}^{\dagger} = \begin{bmatrix} \mathbf{U}_b \\ \mathbf{D}_b \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{11}\mathbf{G}_b^T + \mathbf{A}_{12}\delta\mathbf{G}_b^T \\ \mathbf{A}_{21}\mathbf{G}_b^T + \mathbf{A}_{22}\delta\mathbf{G}_b^T \end{bmatrix} \quad (5)$$

Based on the inversion of 2 by 2 block matrices, we have

$$\begin{aligned}\mathbf{A}_{11} =&(\mathbf{G}_b^T\mathbf{G}_b)^{-1} + (\mathbf{G}_b^T\mathbf{G}_b)^{-1}\mathbf{G}_b^T\delta\mathbf{G}_b\mathbf{R}^{-1}\delta\mathbf{G}_b^T\mathbf{G}_b(\mathbf{G}_b^T\mathbf{G}_b)^{-1}\\ \mathbf{A}_{12} =&-(\mathbf{G}_b^T\mathbf{G}_b)^{-1}\mathbf{G}_b^T\delta\mathbf{G}_b\mathbf{R}^{-1}\\ \mathbf{A}_{21} =&-\mathbf{R}^{-1}\delta\mathbf{G}_b^T\mathbf{G}_b(\mathbf{G}_b^T\mathbf{G}_b)^{-1}\\ \mathbf{A}_{21} =&\mathbf{R}^{-1}\end{aligned} \tag{6}$$

where

$$\mathbf{R} = \delta\mathbf{G}_b^T\delta\mathbf{G}_b - \delta\mathbf{G}_b^T\mathbf{G}_b(\mathbf{G}_b^T\mathbf{G}_b)^{-1}\mathbf{G}_b^T\delta\mathbf{G}_b = \delta\mathbf{G}_b^T(\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger)\delta\mathbf{G}_b \tag{7}$$

with $\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger$ a symmetric and projection matrix, we have

$$\begin{aligned}\mathbf{D}_b =&\mathbf{R}^{-1}\delta\mathbf{G}_b^T - \mathbf{R}^{-1}\delta\mathbf{G}_b^T\mathbf{G}_b\mathbf{G}_b^\dagger\\ =&(\delta\mathbf{G}_b^T(\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger)\delta\mathbf{G}_b)^{-1}\delta\mathbf{G}_b^T(\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger)\\ =&(\delta\mathbf{G}_b^T(\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger)^T(\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger)\delta\mathbf{G}_b)^{-1}\delta\mathbf{G}_b^T(\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger)^T\\ =&((\mathbf{I}-\mathbf{G}_b\mathbf{G}_b^\dagger)\delta\mathbf{G}_b)^\dagger\end{aligned} \tag{8}$$

$$\begin{aligned}\mathbf{U}_b =&\mathbf{G}_b^\dagger + \mathbf{G}_b^\dagger\delta\mathbf{G}_b\mathbf{R}^{-1}\delta\mathbf{G}_b^T\mathbf{G}_b\mathbf{G}_b^\dagger - \mathbf{G}_b^\dagger\delta\mathbf{G}_b\mathbf{R}^{-1}\delta\mathbf{G}_b^T\\ =&\mathbf{G}_b^\dagger - \mathbf{G}_b^\dagger\delta\mathbf{G}_b\mathbf{D}_b\end{aligned} \tag{9}$$

Now we can obtain our Error Minimized Output Basis Functions ELM algorithm for time-variant network (EM-OB). For the sake of presentation clarity, we only add output basis functions one by one in our proposed EM-OB algorithm, which can be easily generalized to the group-by-group node addition means.

Assume we have a set of training data $\{(\mathbf{x}[n], \mathbf{t}[n])\}_{n=1}^N$, the target matrix $\mathbf{T}$, the residual matrix $\mathbf{E}_b = \mathbf{G}_b\mathbf{G}_b^\dagger\mathbf{T} - \mathbf{T}$, the maximum number of output basis functions B_{max}, and the expected learning accuracy ϵ. We get Algorithm 2.

4 Simulation Results

In this section, we compare I-OB and EM-OB with ELM-TV in time-variant MLP and Narendra identification problems. All the data depicted in figures are the average values of 10 trials. All the programs are run in MATLAB 7.8.0 environment with Windows Vista, Intel Core2 Duo CPU P8400 2.26GHz.

4.1 Time-Variant MLP System Identification

The system to be identified here is the same with that in [5,6,8]: a time-variant IIR-buffered MLP with 11 input lines, one 5 neurons hidden layer, one neuron output layer. The input weights and output weights are combination of 3 Chebyshev basis functions; the lengths of the input and output TDLs are equal to 6 and 5 respectively. Note that the output neuron of this system is not linear, both the hidden neurons and output neuron use tangent sigmoid activation function.

A range of output basis functions from 1 to 20 are tested in this scenario. The accuracy performance of EM-OB is quite the same with that of ELM-TV.

Algorithm 2. EM-OB

```
1: Randomly generate input weights and calculate $\mathbf{H}$.
2: Calculate the time-variant hidden layer output matrix $\mathbf{G}_1$:
   $$\mathbf{G}_1 = \begin{bmatrix} f_1[1] \cdot \mathbf{h}[1]^T \\ \vdots \\ f_1[N] \cdot \mathbf{h}[N]^T \end{bmatrix}_{N \times K}$$
3: Calculate the output error $\|\mathbf{E}_1\| = \|\mathbf{G}_1 \mathbf{G}_1^\dagger \mathbf{T} - \mathbf{T}\|$.
4: Let $b = 1$
5: while $b < B_{max}$ and $\|\mathbf{E}_b\| > \epsilon$, do
6:     Add another output basis. The corresponding time-variant hidden layer output
       matrix becomes $\mathbf{G}_{b+1} = [\mathbf{G}_b, \delta\mathbf{G}_b]$, where
       $$\delta\mathbf{G}_b = \begin{bmatrix} f_b[1] \cdot \mathbf{h}[1]^T \\ \vdots \\ f_b[N] \cdot \mathbf{h}[N]^T \end{bmatrix}_{N \times K}$$
7:     Update the output weight $\mathbf{\Omega}$
       $$\mathbf{D}_b = ((\mathbf{I} - \mathbf{G}_b \mathbf{G}_b^\dagger)\delta\mathbf{G}_b)^\dagger$$
       $$\mathbf{U}_b = \mathbf{G}_b^\dagger - \mathbf{G}_b^\dagger \delta\mathbf{G}_b \mathbf{D}_b$$
       $$\mathbf{\Omega}^{(b+1)} = \mathbf{G}_{b+1}^\dagger \mathbf{T} = \begin{bmatrix} \mathbf{U}_b \\ \mathbf{D}_b \end{bmatrix} \mathbf{T}$$
8:     $b = b + 1$
9: end while
```

4.2 Time-Variant Narendra System Identification

The next test identification system is a modified one addressed in [10], adding the coefficients $a[n]$ and $b[n]$ to form a time-variant system, as done in [5,6,8] :

$$y[n] = a[n] \cdot \frac{y[n-1] \cdot y[n-2] \cdot y[n-3] \cdot x[n-1] \cdot (y[n-3] - 1) + x[n]}{1 + b[n] \cdot (y[n-3]^2 + y[n-2]^2)} \quad (10)$$

Simulation results for Narendra system are depicted in Fig 3 and Fig 4. We can conclude from these that I-OB and EM-OB produce more stable results than ELM-TV in Narendra system. Again, same behavior as in the time-invariant case is recognized. The performance comparison in terms of necessary output basis functions between ELM-TV and EM-OB, has also been conducted in these time-variant systems. The target training RMSE(dB) ϵ are set as: -21 for MLP system and -16 for Narendra system. The optimal number of output basis functions for ELM-TV is obtained by trial-and-error. Table 1 displays performance evaluation between ELM-TV and EM-OB, since I-OB is not able to reach the training

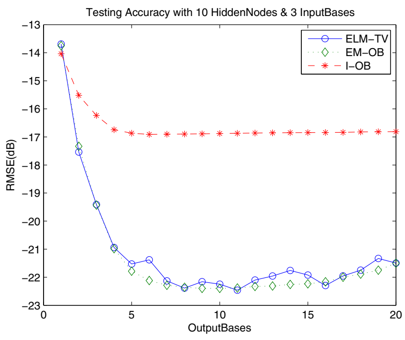


Fig. 1. Testing Accuracy of the three discussed algorithms for MLP System

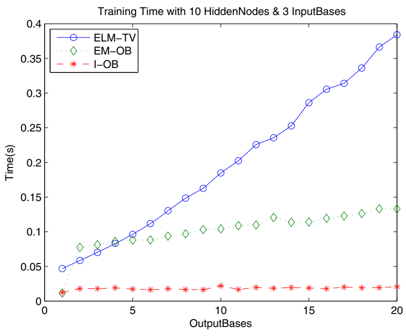


Fig. 2. Training Time of the three discussed algorithms for MLP System

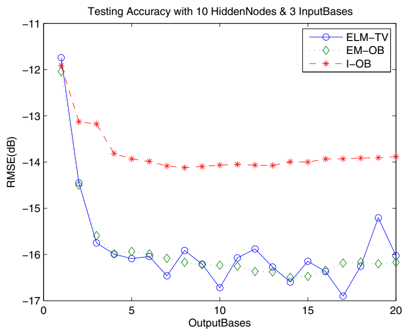


Fig. 3. Testing Accuracy of the three discussed algorithms for Narendra System

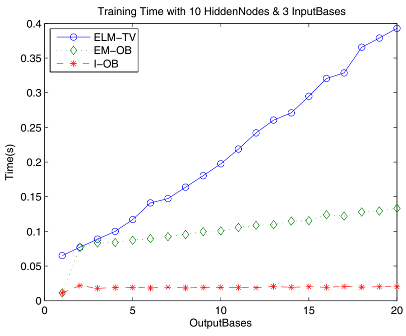


Fig. 4. Training Time of the three discussed algorithms for Narendra System

Table 1. Performance Comparisons for the Number(average of 10 trials) of Output Basis Functions When the Expected Accuracy is Reached

Systems [stop RMSE(dB)]	Algorithms	Training Time (s)	Testing RMSE(dB)	Output Basis Functions
MLP [-21]	ELM-TV	0.1189	-21.63	5.9
	EM-OB	0.0958	-21.17	5.1
Narendra [-16]	ELM-TV	0.1462	-16.07	6.0
	EM-OB	0.0945	-16.24	6.2

goals of them within 20 bases. Focusing on the MLP and Narendra Systems case studies for reasons explained above, results reported in Table 1 prove that EM-OB has similar generalization performance and optimal number of output basis functions attainable with ELM-TV, but at reduced training time. Therefore, EM-OB is able to optimally select the number of output basis functions more efficiently than the trial-and-error approach typically used in common ELM-TV.

5 Conclusions

In this paper, we have proposed two output basis functions incremental Extreme Learning Machine algorithms for Time-Variant Neural Networks training. They are the extensions of the corresponding time-invariant I-ELM and EM-ELM, and they have been addressed as I-OB and EM-OB.

The main advantage of the incremental approach consists in automatically determining the number of output basis functions. Two system identification simulations show that EM-OB approach can significantly reduce the training time, with good generalization performances and similar optimal number of output basis functions comparable to the original ELM.

Moving from results obtained in this paper and in [8], future works are intended to apply a joint ELM-based incremental method for automatically determine the number of hidden nodes and of output basis functions to Time-Variant Neural Networks.

References

1. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: a new learning scheme of feedforward neural networks. In: Proc. IEEE International Joint Conference on Neural Networks, July 25-29, vol. 2, pp. 985–990 (2004)
2. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing 70(1-3), 489–501 (2006)
3. Huang, G., Chen, L., Siew, C.: Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Transactions on Neural Networks 17(4), 879 (2006)
4. Feng, G., Huang, G.B., Lin, Q., Gay, R.: Error minimized extreme learning machine with growth of hidden nodes and incremental learning, vol. 20(8), pp. 1352–1357 (August 2009)
5. Titti, A., Squartini, S., Piazza, F.: A new time-variant neural based approach for nonstationary and non-linear system identification. In: Proc. IEEE International Symposium on Circuits and Systems ISCAS 2005, May 23-26, pp. 5134–5137 (2005)
6. Cingolani, C., Squartini, S., Piazza, F.: An extreme learning machine approach for training time variant neural networks. In: Proc. IEEE Asia Pacific Conference on Circuits and Systems APCCAS 2008, pp. 384–387 (November 2008)
7. Ye, Y., Squartini, S., Piazza, F.: A group selection evolutionary extreme learning machine approach for time-variant neural networks. In: Proocedings of WIRN 2010. IOS Press, Amsterdam (2010)
8. Ye, Y., Squartini, S., Piazza, F.: Incremental-Based Extreme Learning Machine Algorithms for Time-Variant Neural Networks. In: Advanced Intelligent Computing Theories and Applications, pp. 9–16 (2010)
9. Huang, G., Chen, L.: Enhanced random search based incremental extreme learning machine. Neurocomputing 71(16-18), 3460–3468 (2008)
10. Narendra, K., Parthasarathy, K.: Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks 1(1), 4–27 (1990)

Invariant Set and Attractor of Discrete-Time Impulsive Recurrent Neural Networks

Bing Li and Qiankun Song

Department of Mathematics, Chongqing Jiaotong University,
Chongqing 400074, China
{libingcnjy,qiankunsong}@163.com

Abstract. In this paper, we study the invariant set and attractor of the discrete-time impulsive recurrent neural networks (DIRNNs). By using a powerful delay difference inequality and properties of nonnegative matrices, we get some sufficient criteria to determine the invariant set and attractor of DIRNNs. Some examples demonstrate the efficiency.

Keywords: Invariant set, Attractor, Exponential stability, Neural networks.

1 Introduction

Over the past few decades, the recurrent neural networks (RNNs) have attracted more and more attentions because of the rich dynamical behaviors. It plays an important role in a variety of information processing systems such as signal processing, pattern recognition, optimization, model identification and associative memories. Note that, up to now, most RNNs have been assumed to act in a continuous-time manner. However, the discrete-time recurrent neural networks (DRNNs) which is described by difference equation usually appears in the numerical solution of continuous-time networks or it comes to the implementation of continuous-time networks for the sake of computer-based simulation, experimentation, etc. In recently, many researches about discrete-time neural networks, including DRNNs, focus on a variety of dynamical behaviors such as stability, invariant set, attractor and so on [1]- [10].

As we know, pulse phenomena usually exist in our world. The discrete-time impulsive recurrent neural networks (DIRNNs) may exhibit several real world phenomena such as rhythmical beating, merging of solutions and noncontinuous of networks. Then the DIRNNs are emerging as an important area of investigation, since it has much richer dynamical behaviors than networks without impulse. We can find some studies on stability and invariant set of discrete-time dynamic system such as [11, 12]. Unfortunately, up to now, to the best of authors knowledge, there is not enough theories on DIRNNs because of some new difficulties with impulsive effects. Therefore, new techniques and methods on invariant set and attractor should be developed and explored.

For the above reasons, the aim of this paper is developing some new methods and techniques to discuss the invariant set, attractor of DIRNNs. This work is

© Springer-Verlag Berlin Heidelberg 2011

organized as follows. In Section 2, we introduce a general model and some preliminaries. In Section 3, by using an improved difference inequality and properties of nonnegative matrices, we obtain a few sufficient conditions guaranteeing global attracting and exponentially stable. Some numerical examples and simulations are given in Section 4 to show the efficiency of proposed methods.

2 Model and Preliminaries

Throughout this paper, let R and R^+ be the sets of real numbers and nonnegative real numbers, respectively. R^n is the space of n-dimensional real column vectors and $R^{m\times n}$ represents the class of $m \times n$ matrices with real components. For $x \in R^n$, $A \in R^{m\times n}$, we denote $[x]^+ = col\{|x_1|, \cdots, |x_n|\}$, $[A]^+ = (|a_{ij}|)_{m\times n}$. 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 " $\leq$ "(" $>$ "). R^n_+ means a set $R^n_+ = \{x \in R^n | x \geq 0\}$ and $R^{n\times n}_+$ means a set $R^{n\times n}_+ = \{A \in R^{n\times n} | A \geq 0\}$. Z and Z^+ denote the integers set and nonnegative integers set, respectively. $Z[t_1, t_2] \equiv \{t_1, t_1 + 1, \cdots, t_2\}$ where t_1, $t_2 \in Z$ and $t_1 < t_2$. C_τ means the set of all functions $\phi : Z[-\tau, 0] \to R^n$, $\tau \in Z^+$. For any $\phi = col\{\phi_1, \cdots, \phi_n\} \in C_\tau$, we define $[\phi]^+_\tau = col\{||\phi_1||_\tau, \cdots, ||\phi_n||_\tau\}$, in which $||\phi_i||_\tau = \max\limits_{-\tau\leq m\leq 0} |\phi_i(m)|$, $i \in Z[1, n]$.

We consider the following DIRRNs model

$$\begin{cases} x(m+1) = Cx(m) + Af(x(m)) + Bf(x(m-\tau(m))) + J(m), m \neq m_k, \\ x(m_k) = H_k(x(m_k^-)), \quad m = m_k, \\ x(m) = \phi(m), \quad m \in Z[-\tau, 0]. \end{cases} \tag{1}$$

where the neurons $x(m) = col(x_1(m), \cdots, x_n(m)) \in R^n$, the activity functions $f((\cdot) = col(f_1(\cdot), \cdots, f_n(\cdot)) \in R^n$, $m \in Z^+$, the weight of impulse perturbations $H_k(x) = col\{H_{k1}(x), \cdots, H_{kn}(x)\} \in R^n$, $0 < \tau(m) \leq \tau$. The initial string $\phi(m) \in C_\tau$. The fixed impulsive moments m_k satisfy $0 < m_1 < m_2 < \cdots$ and $\lim\limits_{k\to+\infty} m_k = +\infty$. The real valued sequence $x(m, \phi)$ satisfying (1) is called a solution denoted simply by $x(m)$ if no confusion occurs. At first, we assume (1) satisfies the following assumptions
H1. $[f(x)]^+ \leq M[x]^+$, where $M \in R^{n\times n}_+$,
H2. $[J(m)]^+ \leq J$, where $J = col\{J_1, \cdots, J_n\} \in R^n_+$,
H3. $[H_k(x)]^+ \leq R_k[x]^+$, where $R_k \in R^{n\times n}_+$.

The following definitions and lemmas are employed throughout this paper.

Definition 1. *The set $S \subset C_\tau$ is called a positive invariant set of (1), if for any initial string $\phi(m) \in S$, the corresponding solution $x(m, m_0, \phi) \in S$, $\forall m > m_0$.*

Definition 2. *The set $D_1 \subset C_\tau$ is called to be global attracting for the solutions of model (1) if D_1 possesses an open neighborhood D_2 such that for any initial string $\phi \in D_2$,*

$$dist(x(m, \phi), D_1) \longrightarrow 0, \quad as \quad m \longrightarrow +\infty, \tag{2}$$

in which $dist(x, D_1) = \inf_{y \in D_1} d(x, y)$, *and* $d(x, y)$ *is the distance of* x *to* y *in* R^n. *The open set* D_2 *is called to be a basin of attracting of* D_1.

Lemma 1. *[13] If the matrix* $A = \alpha E - P$, *in which* $P \in R_+^{n\times n}$ *and* $\alpha > \rho(P)$, *then* A *is a nonsingular* $\mathcal{M}$*-matrix.*

Lemma 2. *[13] If* A *is a nonsingular* $\mathcal{M}$*-matrix, then* $A^{-1} \in R_+^{n\times n}$.

Lemma 3. *[13] Suppose that* $M \in R_+^{n\times n}$ *and* $\rho(M) < 1$, *then there is a positive vector* $z \in R_+^n$ *such that*

$$(E - M)z > 0. \tag{3}$$

For $M \in R_+^{n\times n}$ with $\rho(M) < 1$, we denote

$$\Omega_\rho(M) = \{z \in R^n | (E - M)z > 0, z > 0\}, \tag{4}$$

which is not empty by Lemma 3.

3 Main Results

In this section, by using an improved difference inequality and properties of nonnegative matrices, we derive some new sufficient conditions guaranteeing positive invariant, global attracting of the DIRNNs model (1).

Lemma 4. *Let* $x(m) \in R_+^n$ *be a vector sequence of real numbers satisfying the inequality as follows:*

$$x(m + 1) \le P_1 x(m) + P_2 x(m - \tau) + I, \quad m > 0. \tag{5}$$

Suppose that $\rho(P_1 + P_2) < 1$, *then* $x(m)$ *satisfies*

$$x(m) \le K e^{-\lambda m} + (E - P_1 - P_2)^{-1} I, \quad \forall m > 0, \tag{6}$$

provided that there exists a constant vector $K \in R_+^n$ *such that the initial string satisfies*

$$\phi(m) \le K e^{-\lambda m} + (E - P_1 - P_2)^{-1} I, \quad \forall m \in Z[-\tau, 0], \tag{7}$$

where the real number $\lambda > 0$ *is determined by the following inequality*

$$(P_1 + P_2 e^{\tau\lambda})K \le e^{-\lambda} K, \tag{8}$$

Proof. The proof is similar to the Lemma 3 in [12]. □

Next, some new criteria on positive invariant set and attractor of (1) will be obtained.

Theorem 1. *Assume that* $\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+) < 1$ *and* $\rho(R_k) \le 1$, *then the set* $S = \{\phi(m) \in C_\tau | [\phi(m)]_\tau^+ \le T\}$ *is a positive invariant set of (1), where* $T = (E - ([C]^+ + [A]^+ M]^+ + [B]^+ M]^+))^{-1} J$.

Proof. For any initial string $\phi(m) \in S \subset C_\tau$, we have

$$[\phi(m)]_\tau^+ \leq T, \quad -\tau \leq m \leq 0. \tag{9}$$

By Lemma 4, we obtain that the corresponding solution $x(m, \phi)$ satisfies

$$[x(m)]^+ \leq T, \quad 0 \leq m < m_1. \tag{10}$$

When $m = m_1$, we have

$$\begin{aligned} [x(m_1)]^+ &= [H_1(x(m_1^-))]^+ \leq R_1[x(m_1^-)]^+ \\ &\leq \rho(R_1)[x(m_1^-)]^+ \\ &\leq T. \end{aligned} \tag{11}$$

According to Lemma 4 again, we obtain

$$[x(m)]^+ \leq T, \quad m_1 \leq m < m_2. \tag{12}$$

By mathematical induction, we can get

$$[x(m)]^+ \leq T, \quad \forall m > 0. \tag{13}$$

We complete the proof. □

Theorem 2. *Suppose that model (1) satisfies assumptions as follows*
(A1). $\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+) < 1$,
(A2). there exist positive sequences γ_k and σ_k in which $\gamma_k \geq \max\{1, \rho(R_k)\}$, $\sigma_k \geq \max\{1, \rho(R_k)\}$, such that $R_k K \leq \gamma_k K$ for $K \in \Omega_\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+)$ and $R_k T \leq \sigma_k T$,
(A3). there are two positive numbers γ and σ such that

$$\frac{\ln \gamma_k}{m_k - m_{k-1}} \leq \gamma < \lambda, \quad \prod_{k=1}^{+\infty} \sigma_k \leq \sigma, \tag{14}$$

where λ is determined by

$$([C]^+ + [A]^+[M]^+ + [B]^+[M]^+ e^{\tau\lambda})K \leq e^{-\lambda}K,$$

then the set $D = \{\phi(m) \in C_\tau | [\phi(m)]_\tau^+ \leq \sigma T\}$ is a global attracting set of (1).

Proof. Considering DIRNNs model (1) under (H1)-(H3), we can transform (1) to

$$\begin{cases} [x(m+1)]^+ \leq ([C]^+ + [A]^+[M]^+)[x(m)]^+ \\ \qquad\qquad\quad +([B]^+[M]^+)[x(m-\tau(m))]^+ + J, \\ [x(m_k)]^+ \quad = [H_k(x(m_k^-))]^+ \leq R_k[x(m_k^-)]^+, \quad m = m_k. \end{cases} \tag{15}$$

The initial string comes to be

$$[\phi(m)]_\tau^+ = col\{||\phi_1(m)||_\tau, \cdots, ||\phi_n(m)||_\tau\}, \quad \forall m \in Z[-\tau, 0]. \tag{16}$$

Since $\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+) < 1$ and $[C]^+ + [A]^+[M]^+ + [B]^+[M]^+ \in R_+^{n\times n}$, then, from Lemma 3, we know that there must be a positive vector $z \in \Omega_\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+)$ such that $(E - ([C]^+ + [A]^+[M]^+ + [B]^+[M]^+))z > 0$. Combining with (16), we denote

$$K = \max_{i\in Z[1,n]}\{\frac{||\phi_i(m)||_\tau}{z_i}\}z, \tag{17}$$

Obviously, we get $K \in \Omega_\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+)$.

It is easy to see that the initial string $\phi(m)$ satisfies

$$[\phi(m)]_\tau^+ \leq K, \quad \forall m \in Z[-\tau, 0]. \tag{18}$$

From (A1), it is not difficult to choose a positive scalar λ such that $([C]^+ + [A]^+[M]^+ + [B]^+[M]^+e^{\tau\lambda})K \leq e^{-\lambda}K$ by continuity. Meantime, we know $T = (E-([C]^+ + [A]^+M]^+ + [B]^+M]^+))^{-1}J > 0$. Based on these, we can easily obtain

$$[\phi(m)]_\tau^+ \leq Ke^{-\lambda m} + T, \quad \forall m \in Z[-\tau, 0]. \tag{19}$$

We denote $\gamma_0 = \sigma_0 = 1$ and $m_0 = 0$. By employing Lemma 4, (15) and (19), we derive that

$$[x(m)]^+ \leq \gamma_0 Ke^{-\lambda m} + \sigma_0 T, \quad m_0 \leq m < m_1. \tag{20}$$

Suppose that for all $\nu = 1, \cdots, k$, the inequality as follows hold

$$[x(m)]^+ \leq \gamma_0\gamma_1\cdots\gamma_{\nu-1}Ke^{-\lambda m} + \sigma_0\sigma_1\cdots\sigma_{\nu-1}T, \quad m_{\nu-1} \leq m < m_\nu. \tag{21}$$

Next, we will show the conclusion is still correct when $\nu = k+1$. Considering $m = m_k$, by the second inequality in (15) and (21), we can deduce

$$\begin{aligned}[x(m_k)]^+ &= [H_k(x(m_k^-))]^+ \leq R_k[x(m_k^-)]^+ \\ &\leq R_k\{\gamma_0\gamma_1\cdots\gamma_{k-1}Ke^{-\lambda m_k} + \sigma_0\sigma_1\cdots\sigma_{k-1}T\} \\ &\leq \gamma_0\gamma_1\cdots\gamma_{k-1}(R_kK)e^{-\lambda m_k} + \sigma_0\sigma_1\cdots\sigma_{k-1}(R_kT).\end{aligned} \tag{22}$$

Employing (A2), we can deduce that (22) changes into

$$[x(m_k)]^+ \leq \gamma_0\gamma_1\cdots\gamma_{k-1}\gamma_kKe^{-\lambda m_k} + \sigma_0\sigma_1\cdots\sigma_{k-1}\sigma_kT \tag{23}$$

Noting that $\gamma_k \geq \max\{1, \rho(R_k)\}$, $\sigma_k \geq \max\{1, \rho(R_k)\}$, from (21) and (23) we can show that

$$[x(m)]^+ \leq \gamma_0\gamma_1\cdots\gamma_kKe^{-\lambda m} + \sigma_0\sigma_1\cdots\sigma_kT, \quad m_k - \tau \leq m \leq m_k. \tag{24}$$

And according to (15), we get

$$\begin{aligned}[x(m+1)]^+ \leq{}& ([C]^+ + [A]^+[M]^+)[x(m)]^+ + ([B]^+M]^+)[x(m-\tau(m))]^+ \\ &+\sigma_0\sigma_1\cdots\sigma_kI, \quad m_k < m < m_{k+1},\end{aligned} \tag{25}$$

Combining (24) with (25) and applying Lemma 4, we can get

$$[x(m)]^+ \leq \gamma_0\gamma_1\cdots\gamma_k K e^{-\lambda m} + \sigma_0\sigma_1\cdots\sigma_k T, \quad m_k \leq m < m_{k+1}. \tag{26}$$

(26) shows that the conclusion (21) is right at $\nu = k+1$. By the mathematic induction, we conclude that $\forall k \in Z^+$

$$[x(m)]^+ \leq \gamma_0\gamma_1\cdots\gamma_k K e^{-\lambda m} + \sigma_0\sigma_1\cdots\sigma_k T, \quad m_k \leq m < m_{k+1}. \tag{27}$$

Since $\frac{\ln \gamma_k}{m_k - m_{k-1}} \leq \gamma < \lambda$, it means that $\gamma_k \leq e^{\gamma(m_k - m_{k-1})}$. Furthermore, because $\prod\limits_{k=1}^{+\infty} \sigma_k \leq \sigma$, we obtain that $\forall k \in Z^+$

$$\begin{aligned}[x(m)]^+ &\leq \gamma_0\gamma_1\cdots\gamma_k K e^{-\lambda m} + \sigma_0\sigma_1\cdots\sigma_k T \\ &\leq e^{\gamma m_k} e^{-\lambda m} K + \sigma T \\ &\leq e^{-(\lambda-\gamma)m} K + \sigma T, \quad m_k \leq m < m_{k+1}.\end{aligned} \tag{28}$$

(28) indicates that

$$[x(m)]^+ \leq e^{-(\lambda-\gamma)m} K + \sigma T, \quad \forall m \geq 0. \tag{29}$$

Let $m \to +\infty$ in both sides of (29). We can deduce

$$\lim_{m\to+\infty} [x(m)]^+ \leq \sigma T \tag{30}$$

By Definition 2, we know that the $D = \{\phi(m) \in C_\tau | [\phi(m)]_\tau^+ \leq \sigma T\}$ is a global attracting set. We complete the proof. □

4 Illustrative Examples

We consider the following DIRNNs

$$y(m+1) = Cy(m) + Af(y(m)) + Bf(y(m-1)) + J(m), \quad m \neq m_k, \tag{31}$$

with impulse perturbation

$$y(m_k) = R(k)y(m_k^-), \quad m = m_k, \tag{32}$$

in which $y = col\{y_1, y_2\}$, $f(\cdot) = col\{f_1, f_2\}$, $J(m) = col\{J_1(m), J_2(m)\}$, $R(k) = diag\{R_1(k), R_2(k)\}$.

CASE 1. We choose $f_1(y_1) = y_1 \sin(y_1)$, $f_2(y_2) = |y_2|$, $J_1(m) = \sin(m)$, $J_2(m) = \cos(m)$ and $R_1(k) = R_2(k) = e^{\frac{1}{(\pi k)^2}}$, $m_k = m_{k-1} + k$, $C = \begin{pmatrix} \frac{1}{4} & 0 \\ 0 & \frac{1}{5} \end{pmatrix}$, $A = \begin{pmatrix} \frac{1}{5} & \frac{1}{10} \\ \frac{1}{6} & \frac{1}{8} \end{pmatrix}$, $B = \begin{pmatrix} -\frac{1}{6} & \frac{1}{8} \\ -\frac{1}{3} & \frac{1}{10} \end{pmatrix}$.

It is easy to see that there is a positive matrix $M = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, such that $[f(y)]^+ \leq M[y]^+$. We derive that $R_k = diag\{e^{\frac{1}{(\pi k)^2}}, e^{\frac{1}{(\pi k)^2}}\}$ and $J = col(1,1)$.

We can easily observe that $\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+) = 0.8697 < 1$.

Let $\gamma_k = \sigma_k = e^{\frac{1}{(\pi k)^2}}$, so (A2) hold. Taking $\lambda = 0.4$, $K = col\{1, 1\}$, we can deduce that $([C]^+ + [A]^+[M]^+ + [B]^+[M]^+e^{\lambda})K \le e^{-\lambda}K$. There exist $\gamma = 0.1013$ and $\sigma = e^{\frac{1}{6}}$ such that $\frac{\ln\gamma_k}{m_k - m_{k-1}} \le 0.1013 < 0.4$ and $\prod_k \sigma_k \le e^{\frac{1}{6}}$. So (A3) hold.

Clearly, by using Theorem 2, we can obtain that the set

$$D = \{\begin{pmatrix} \phi_1(m) \\ \phi_2(m) \end{pmatrix} \in R^2 | \max_{-1\le m\le 0} |\phi_1(m)| \le e^{\frac{1}{6}}7.413, \max_{-1\le m\le 0} |\phi_2(m)| \le e^{\frac{1}{6}}8.185\}$$

is the global attracting set.

Figure 1 illustrates the dynamical behaviors of (31) and (32) under Case 1.

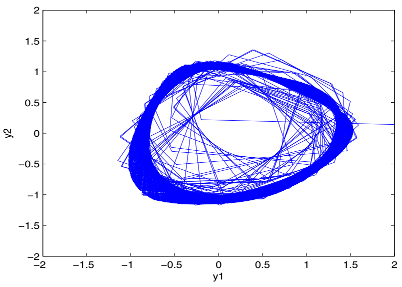


Fig. 1. State trajectory under case 1

CASE 2. We consider $f_1(y) = \frac{1}{5}(y_1 + |y_2|)$, $f_2(y) = \frac{1}{5}(y_1 \sin(y_2) + y_2)$, $I_1(m) = \cos(m)$, $I_2(m) = \frac{1}{2} + (-1)^m\frac{1}{3}$ and $R_1(k) = R_2(k) = e^{\frac{1}{2^k}}$, $m_k = m_{k-1} + k$, $C = \begin{pmatrix} \frac{1}{8} & 0 \\ 0 & \frac{1}{9} \end{pmatrix}$, $A = \begin{pmatrix} \frac{1}{9} & -\frac{1}{10} \\ -\frac{1}{9} & -\frac{1}{10} \end{pmatrix}$, $B = \begin{pmatrix} -\frac{1}{8} & \frac{1}{10} \\ \frac{1}{9} & -\frac{1}{9} \end{pmatrix}$.

It is easy to count out $M = \begin{pmatrix} \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} \end{pmatrix}$, $\rho([C]^+ + [A]^+[M]^+ + [B]^+[M]^+) = 0.2922 < 1$. We can choose $\lambda = 0.5$, $K = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, such that $([C]^+ + [A]^+[M]^+ + [B]^+[M]^+e^{\lambda})K \le e^{-\lambda}K$, $T = (E - ([C]^+ + [A]^+[M]^+ + [B]^+[M]^+))^{-1}J = \begin{pmatrix} 1.4012 \\ 1.1902 \end{pmatrix} > 0$.

Considering $\gamma_k = \sigma_k = e^{\frac{1}{2^k}}$, we get $\gamma = 0.25 < \lambda = 0.5$ and $\sigma = 1 \ge \prod_{k=1}^{\infty} \sigma_k$. By Theorem 2, the set

$$D = \{\begin{pmatrix} \phi_1(m) \\ \phi_2(m) \end{pmatrix} \in R^2 || \phi_1(m)| \le 1.4012, |\phi_2(m)| \le 1.1902\}$$

is the global attracting set.

In Figure 2, we can see the simulation result under Case 2.

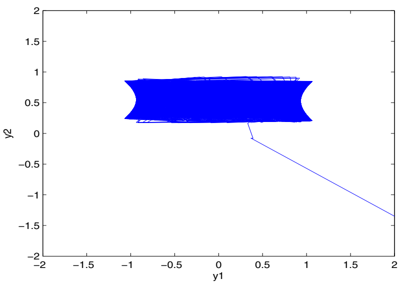


Fig. 2. State trajectory under case 2

Remark 1. Obviously, the results proposed in [2] are invalid because of the existence of impulse. However, our results can go into effect with or without impulse. Moreover, it is easy to see that we can still get the global attracting set even though the weights of impulsive perturbations do not satisfy the assumption $\rho\{R(K)\} < 1$ proposed in [12]. Hence, our results are less conservative than those in [2] and [12].

5 Conclusions

We studied the invariant set and attractor of the discrete-time impulsive recurrent neural networks (DIRNNs) in this paper. By introducing an improved delay difference inequality, we obtained some sufficient criteria to determine the invariant set and attractor of DIRNNs. The examples in Section 4 demonstrate the power of our methods. Compared with some previous results, our results are less conservative.

Acknowledgments. This work was supported by the National Natural Science Foundation of China under Grants 60974132 and 10671133.

References

1. Zhang, H., Chen, L.S.: Asymptotic Behavior of Discrete Solutions to Delayed Neural Networks with Impulses. Neurocomputing 71, 1032–1038 (2008)
2. Liao, X.X., Luo, Q., Zeng, Z.G.: Positive Invariant and Global Exponential Attractive Sets of Neural Networks with Time-varying Delays. Neurocomputing 71, 513–518 (2008)
3. Song, Q.K., Cao, J.D.: Dynamical Behaviors of Discrete-time Fuzzy Cellular Neural Networks with Variable Delays and Impulses. Journal of the Franklin Institute 345, 39–59 (2008)
4. Song, Q.K., Wang, Z.D.: A Delay-dependent LMI Approach to Dynamics Analysis of Discrete-time Recurrent Neural Networks with time-varying Delays. Physics Letters A 368, 134–145 (2007)
5. Liu, Y.R., Wang, Z.D., Serrano, A., Liu, X.H.: Discrete-time Recurrent Neural Networks with Time-varying Delays: Exponential stability Analysis. Physics Letters A 362, 480–488 (2007)

6. Yu, J.J., Zhang, K.J., Fei, S.M.: Exponential Stability Criteria for Discrete-time Recurrent Neural Networks with Time-varying Delay. Nonlinear Analysis: Real World Applications 11, 207–216 (2010)
7. Song, C.W., Gao, H.J., Zheng, W.X.: A New Approach to Stability Analysis of Discrete-time Recurrent Neural Networks with Time-varying Delay. Neurocomputing 72, 2563–2568 (2009)
8. Zhang, B.Y., Xu, S.Y., Zou, Y.: Improved Delay-dependent Exponential Stability Criteria for Discrete-time Recurrent Neural Networks with Time-varying Delays. Neurocomputing 72, 321–330 (2008)
9. Zhu, X.L., Wang, Y.Y., Yang, G.H.: New Delay-dependent Stability Results for Discrete-time Recurrent Neural Networks with Time-varying Delay. Neurocomputing 72, 3376–3383 (2009)
10. Wu, Z.G., Su, H.Y., Chu, J., Zhou, W.N.: New Results on Robust Exponential Stability for Discrete Recurrent Neural Networks with Time-varying Delays. Neurocomputing 72, 3337–3342 (2009)
11. Zhu, W., Xu, D.Y., Yang, Z.C.: Global Exponential Stability of Impulsive Delay Difference Equation. Applied Mathematics and Computation 181, 65–72 (2006)
12. Zhu, W.: Invariant and Attracting Sets of Impulsive Delay Difference Equations with Continuous Variables. Computers and Mathematics with Applications 55, 2732–2739 (2008)
13. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1985)

Optimal Control for Boiler Combustion System Based on Iterative Heuristic Dynamic Programming[*]

Bilian Liao, Kui Peng, Shaojian Song, and Xiaofeng Lin

School of Electrical Engineering, Guangxi University,
Nanning Guangxi 530004, China

Abstract. Boiler combustion system is a complex nonlinear system which has characteristic of strong coupling and strong-disturbance. It is hard to build accurate mathematical model and achieve optimal control for it. In this paper, radial basis function (RBF) neural network model for boiler combustion system is built based on data driven method firstly, then performing the optimal control of the boiler combustion system via the iterative heuristic dynamic programming (HDP) algorithm, and improving the initial weights of neural network and the utility function. Finally compared with the traditional HDP algorithm in Matlab. The result shows that the optimization algorithm of the iteration HDP based on the RBF neural network gets better in overshoot, convergence speed, steady state error, adaptability and robustness.

Keywords: optimal control; iteration HDP; RBF neural network; boiler combustion system.

1 Introduction

Boiler combustion system is an important section in the boiler system, the main steam pressure is an important monitoring parameter of boiler combustion system which is to ensure the safe operation of boiler and the balance energy between boiler and load.

To ensure the safe operation of boiler and the energy balance between the boiler and load, many experts in the optimization of combustion system have done a lot of scientific researches, and develop their corresponding optimization equipments. Some European and American companies develop some optimization softwares, such as NeuSIGHT system is an boiler combustion control system, which uses the artificial intelligence, neural network technology [1]. Li Zhi *et a*l employ boiler combustion optimization with based online technical of the identification, and the convergence speed is improved [2]. Yan Gao *et al* implement the optimal control of

[*] "This work was supported by the Natural Science Foundation of China under Grant 60964002; the Natural Science Foundation of Guangxi Province of China under Grant 0991057; the Science & Research Foundation of Educational Commission of Guangxi Province of China under Grant 200808ms003."

© Springer-Verlag Berlin Heidelberg 2011

boiler combustion system with expert system[3]. Peihong Wang *et al* study the boiler combustion optimizing by the artificial intelligence technology [4]. These studies have played a very important role in guiding the boiler combustion, but remain some disadvantages such as slow response, poor robustness, slow convergence speed and so on. In 1977, Werbos proposed a new optimization method—Adaptive Dynamic Programming (ADP), this method is very suitable to be used in hybrid, nonlinear and non-stationary environment. Huaguang Zhang, Qinglai Wei proposed the greedy iterative algorithm which is used to solve a class of discrete time nonlinear systems and a new infinite time optimal tracking control problem[5]. Qinglai Wei proposed iteration HDP algorithm which is implemented mainly by BP neural network and this method has a good convergence speed and stability[6]. The RBF neural network has advantages such as simple structure, fast training and learning speed, and can approximate any nonlinear function than BP neural network. Therefore, in this paper, boiler combustion system model is built by the RBF neural network based on the data, at the same time design the controller of boiler combustion system with RBF neural network of iterative HDP algorithm and compared with traditional HDP algorithm.

2 Adaptive Dynamic Programming (ADP)

For nonlinear systems, stability is only a bare minimum requirement in a system design. Ensuring optimality guarantees the stability of the nonlinear system. Dynamic programming is a very useful tool to solve optimization and optimal control problems by employing the principle of optimality.

2.1 Dynamic Programming

In order to solve optimization and optimal control problems, dynamic programming is proposed [7]. There are several fields about the dynamic programming, such as multiple-input and multiple-output systems, linear systems and nonlinear systems.

For nonlinear discrete-time:

$$x(k+1) = f[x(k), u(k)], k = 0, 1, 2... \tag{1}$$

Where ***x(k)*** represents the system state vector, ***u(k)*** is the control action, *f* is the system function. Suppose that one associates with this system the performance index (or cost)

$$J[x(i), i] = \sum_{k=i}^{\infty} r^{k-i} U[x(k), u(k), k] \tag{2}$$

According to Bellman equation, the optimal cost from time *k* is equal to

$$J^*(x(k)) = \min_{u(k)} \{U(x(k), u(k)) + \gamma J^*(x(k+1))\} \tag{3}$$

The optimal control at time *k* is the ***u(k)***, which achieves this minimum,

$$u^*(k) = \arg\min_{u(k)}\{U(x(k), u(k)) + \gamma J^*(x(k+1))\} \tag{4}$$

However, for a practical dynamic programming problem, it is often computationally untenable to run true dynamic programming due to the backward numerical process required for its solutions, as a result of the well-known "curse of dimensionality"[8]. In order to overcome the curse of dimensionality problem of dynamic programming, adaptive dynamic programming is proposed.

2.2 Adaptive Dynamic Programming

In 1977, Werbos introduced an approach for ADP that was later called adaptive critic designs (ACDs)[9]. To implement the ADP algorithm, Werbos proposed "approximate dynamic programming" formulations. Adaptive dynamic programming has two basic versions which are heuristic dynamic programming (HDP) and dual heuristic programming (DHP) [10].

HDP is the most basic and widely applied structure of ADP. HDP is a method for estimating the cost function. Estimating the cost function for a given policy only requires samples from the instantaneous utility function *U*, while models of the environment and the instantaneous reward are needed to find the cost function corresponding to the optimal policy[10]. The structure of HDP is shown in Fig. 1.

2.3 The Iterative HDP Algorithm

The structure of iterative HDP is similar to the conventional HDP. For the analysis of convergence and stability, Qinglai Wei has been the strict mathematical proof in his doctoral thesis. Consider the following nonaffine nonlinear system:

$$x(k+1) = f(x_k, u_k) \tag{5}$$

According to equation 3, we can obtain the derivative of ***u***

$$\frac{\partial J^*(x_k)}{\partial u_k} = \frac{\partial x_k^T Q x_k + u_k^T R u_k}{\partial u_k} + \frac{\partial x_{k+1}}{\partial x_k}\frac{\partial J^*(x_{k+1})}{\partial x_{k+1}} = 0$$
$$u^*(x_k) = \frac{1}{2}R^{-1}\left(\frac{\partial x_{k+1}}{\partial u_k}\right)^T \frac{\partial J^*(x_{k+1})}{\partial x_{k+1}} \tag{6}$$

Combining equation 3 and 6, the equation of discrete-time HJB can be derived, and then the explicit optimal control expression u^* is obtained by solving the HJB equation. The schematic of iteration HDP algorithm as follow:

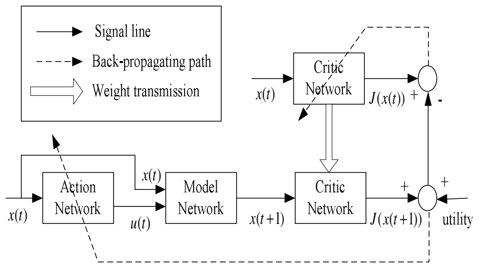
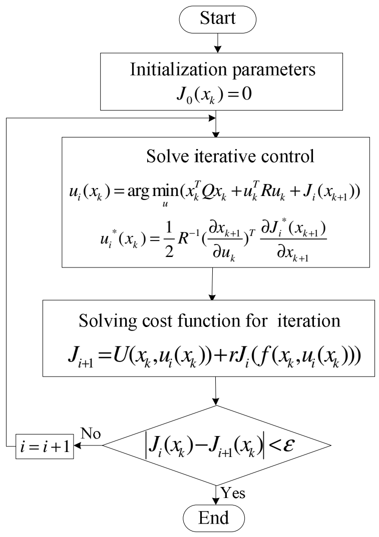


Fig. 1. The structure diagram of the HDP



Start
Initialization parameters
$J_0(x_k)=0$
Solve iterative control
$u_i(x_k)=\arg\min_u(x_k^TQx_k+u_k^TRu_k+J_i(x_{k+1}))$
$u_i^*(x_k)=\frac{1}{2}R^{-1}(\frac{\partial x_{k+1}}{\partial u_k})^T\frac{\partial J_i^*(x_{k+1})}{\partial x_{k+1}}$
Solving cost function for iteration
$J_{i+1}=U(x_k,u_i(x_k))+rJ_i(f(x_k,u_i(x_k)))$
$|J_i(x_k)-J_{i+1}(x_k)|<\varepsilon$
No
$i=i+1$
Yes
End

Fig. 2. Iteration HDP algorithm process

3 Building the Neural Network Model of the Boiler Combustion

3.1 Neural Network Model of the Boiler Combustion

In this paper, model of the boiler combustion system is built by RBF neural network. For a boiler combustion system, the control variables are: the fuel quantity (B), supplied air rate (V) and air input amount rate (G), the state variable is: the main steam pressure (PT). As shown in Fig.3, sm represents the input of the network model , $sm=[x(t)\ u(t)]$, where $x(t)$ is the state variable at time *t*, $u(t)$ is the control variables at time *t*, $x(t+1)$ is the actual output value of the model network, $\overline{x(t+1)}$ is the desired output value, e_m is error of the neural network model. In this paper, the structure of the model of the boiler combustion system is 4-60-1.

The transfer function of RBF neural network is Gaussian function:

1) Initialize the network parameters, the initial input is $sm=[x(t)\ u(t)]$, the center of Gaussian function is c_m, the variance of Gaussian function is b_m, the weights from hidden layer to output layer are w_m; the mapping from input to output is:

$$mh=\exp[-\frac{\|sm-c_{mk}\|^2}{b_{mk}^{\ 2}}],\text{k=1,2}\cdots 60 \quad x(t+1)=w_m mh \tag{7}$$

2) Define the error of the model network $e_m=\overline{x(t+1)}-x(t+1)$, $E=0.5e_m^{\ 2}$.
3) Using the gradient descent method adjust the RBF neural network parameters.

So the gradient-based weight updating rule for the model network is given by

$$w_m = w_m - \eta e_m \frac{\partial x(t+1)}{\partial w_m} = \eta e_m mh \tag{8}$$

The center of Gaussian function updating rule for the model network is given by:

$$c_m = c_m + \eta e_m \frac{\partial x(t+1)}{\partial mh}\frac{\partial mh}{\partial c_m} \tag{9}$$

The variance of Gaussian function updating rule for the model network is given by:

$$b_m = b_m + \eta e_m \frac{\partial x(t+1)}{\partial mh}\frac{\partial mh}{\partial b_m} \tag{10}$$

Where the η =0.05 is learning rate.

3.2 Model Testing and Results Analysis of Generalization

The data-based RBF neural network model has good generalization ability from the simulation result. This is because RBF neural network has many advantages such as: fast learning, a wide range of data fusion, and the ability of parallel processing of data. Thus, the model of boiler combustion system has good generalization ability and accuracy based on RBF neural network.

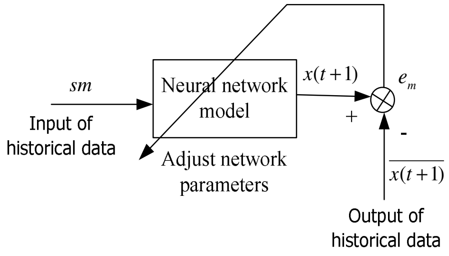


Fig. 3. The idea of modeling of boiler

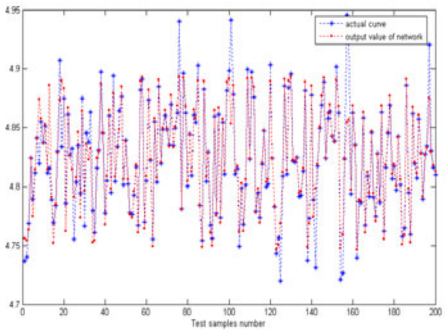


Fig. 4. Generalization ability of the model

4 Neural Network Implementation of Iteration HDP

In this paper, we use the iteration HDP to design the controller of the boiler combustion system based on RBF network. This control objective is to minimize cost of the output of the critic network, then the optimal control is a linear state feedback $u(x_x) = f(x_k) * x_k$, therefore, the training goal of iteration HDP is to find the feedback function $f(x_k)$.

For the conventional RBF neural network, assuming that the input is ***x***, the number of hidden layer neurons is 60,that is because the RBF neural network has a strong classification, and this controller is trained 1000 times, so in order to get a better clustering, the number of the center of the critic network and action network are 60.The center and width of the Gaussian function is the ***c*** and ***b***, the weight matrix between the hidden layer and output layer is denoted by *W*, then the output of neural network is represented by:

$$F(x,W) = W^T \exp(-\frac{\|x-c\|^2}{b^2}) \tag{11}$$

4.1 The Model Network

The model network is to approximate the system dynamic and it should be trained before the implementation of the iterative ADP algorithm. The update rule of the model network is adopted as gradient decent method. After the model network is trained, its weights are kept unchanged.

4.2 The Critic Network

The structure of critic network is: 1-60-1. The critic network is used to approximate the performance index function $J^i(x_k)$. The output of the critic network is denoted as

$$\hat{J}^i(x(k)) = W_c^T f(x(k)) \tag{12}$$

The target function can be written as

$$J^{i+1}(x(k)) = U^i(x(k),u(x(k))) + \gamma \hat{J}^i(x(k+1)) \tag{13}$$

Where $U^i(x(k),u(x(k)))$ is the utility function, γ ($0 < \gamma \leq 1$) is discount factor.

Then we define the error function for the critic network

$$e_c(x(k)) = \hat{J}^{i+1}(x(k)) - J^{i+1}(x(k)) \quad E_c = \frac{1}{2} e_c(x(k))^T e_c(x(k)) \tag{14}$$

So the gradient-based the center of Gaussian function updating rule for the critic network is given by

$$c_{ci+1} = c_{ci} - \sum_{i=1}^{L}\sum_{j=1}^{N} e_{cj} G(-\frac{\|x(k)-c_{ci}\|^2}{b_{ci}^{\ 2}})(x(k)-c_{ci}) \tag{15}$$

The centers of the Gaussian function updating rule for the critic network is given by

$$b_{ci+1} = b_{ci} - \sum_{i=1}^{L}\sum_{j=1}^{N} e_{cj} G(-\frac{\|x(k) - c_{ci}\|^2}{b_{ci}^3}) \tag{16}$$

The gradient-based weight updating rule for the critic network is given by

$$w_{ci+1} = w_{ci} - \sum_{j=1}^{N} e_{cj} G(-\frac{\|x(k) - c_c\|^2}{b_c^2}) \tag{17}$$

Where L represents the number of the center, N represents the number of output node l_c =0.05 is the learning rate of critic network.

4.3 The Action Network

The goal of action network is to find the optimal control signals, so as to minimize the cost-to-go function $J(x(k))$.

$$\frac{\partial J(x(k))}{\partial u(k)} = \frac{\partial U}{\partial u(k)} + r\frac{\partial J(k+1)}{\partial u(k)} = 0 \tag{18}$$

The optimal control is

$$u^*(k) = \frac{1}{2} R^{-1} (\frac{\partial x(k+1)}{\partial u(k)})^T \frac{\partial J^*(x(k+1))}{\partial x(k+1)} \tag{19}$$

In the action network the state $\boldsymbol{x}(k)$ is used as input to create the optimal control as the output of the network. The output can be formulated as

$$\hat{u}^i(k) = W_a^T f(x(k)) \tag{20}$$

The error function of action network is defined as:

$$e_a = \hat{u}^i(k) - u^*(k) \qquad E_a = \frac{1}{2} e_a^T e_a \tag{21}$$

The weights updating algorithm is similar to the one for the critic network. By the gradient descent rule, we can obtain

$$w_{ai+1}(k) = w_{ai}(k) - l_a e_a \frac{\partial e_a}{\partial w_a} \tag{22}$$

$$c_{ai+1}(k) = c_{ai}(k) - l_a e_a \frac{\partial e_a}{\partial c_a} \tag{23}$$

$$b_{ai+1}(k) = b_{ai}(k) - l_a e_a \frac{\partial e_a}{\partial b_a} \tag{24}$$

Where l_a =0.02 is the learning rate of action network, r is the discount factor.

4.4 Improvement of the Training Strategy

The initial value of the neural network is randomly generated, so it is difficult to ensure the stability. Therefore, this paper based on the saved data, which is the best training results of a number of times in the course of training, then we can directly load the last saved data to train the neural network.

4.5 Improvement of Utility Function

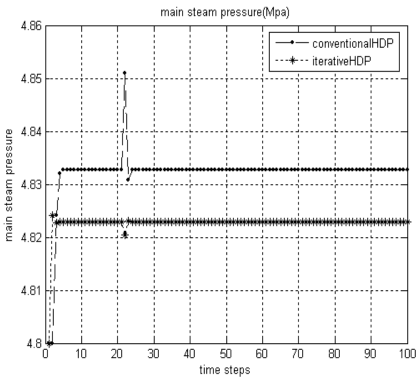
Traditional utility function is defined as a quadratic about state variables and control variables. When the control objective is given, the utility function can be defined as the way with a tracking error.

$$Z = (x_k - \bar{x}) \quad P = (u_k - u_i^*(x_k)) \quad U = Z^T QZ + P^T RP \tag{25}$$

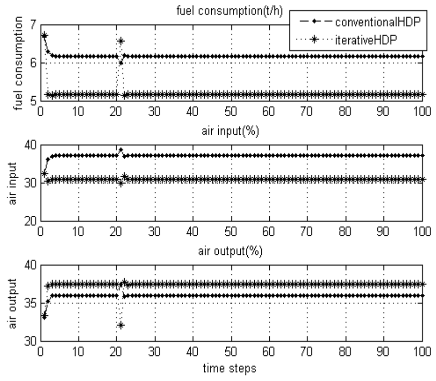
Where $\bar{x}$ represents control objectives, Q is positive definite quadratic form, R is semi-positive definite quadratic form.

5 Simulation and Testing

The initial condition is *x(k)* = 0.5, the model network has been trained offline. The control objective of the main steam pressure is maintained at 4.82MPa nearby. In this paper iteration HDP algorithm and conventional HDP algorithm are programmed, and compared the result of them in the Matlab environment. The control effects of iterative HDP and the traditional HDP are shown in Fig.5.



(a) Result of main steam pressure



(b) Result of optimal control

Fig. 5. Shows the result of experimental

The experimental results show that, the controller which is based on RBF neural network of iterative HDP algorithm not only features its strong robustness, high speed of convergence and high control precision, but also in terms of energy consumption and combustion efficiency is much better than conventional HDP. The coal consumption of the two controllers is: 5.16t/h and 6.17t/h based on iterative HDP algorithm and conventional HDP algorithm.

6 Conclusion

In this paper, first RBF neural network model of boiler combustion system is built .RBF neural network have a good approximation for nonlinear systems. Experiment shows that: the RBF neural network model has good generalization ability. Then optimal control system of the boiler combustion is designed the by iterative HDP method. The simulation result shows that: the controller of boiler combustion system has many advantages of strong robustness, fast convergence speed and high control precision based on RBF neural network of the iterative HDP algorithm. So the controller not only meets the security requirement of the boiler combustion system in industrial production, but also achieves the target of low consumption and low pollution.

References

1. Kong, L., Zhang, Y.: Boiler Combustion Optimization Technology Research Development. J. Boiler Technology 39(5), 33–36 (2008)
2. Li, Z., Cai, J., et al.: Optimization system of power plant boiler combustion based on neural network. M. China Power (2004)
3. Gao, Y., et al.: Optimal control of boiler combustion system based on expert system. J. Power Automation Equipment. 25(5), 27–29 (2005)
4. Wang, P., Li, L.: Study on optimal control of the boiler combustion system by artificial intelligence technology. J. The Chinese Society for Electrical Engineering 24(4), 184–188 (2004)
5. Zhang, H., Wei, Q.: A Novel Infinite-Time Optimal Tracking Control Scheme for a Class of Discrete-Time Nonlinear Systems via the Greedy HDP Iteration Algorithm. J. IEEE Transactions on systems MAN, and Cybernetics- Partb: Cybernetics 38(4), 937–942 (2008)
6. Wei, Q., Zhang, H.: An Optimal Control Scheme for a Class of Discrete-Time Nonlinear Systems with Time Delays Using Adaptive Dynamic Programming. J. Acta Automatica Sinica (2008)
7. Li, G., et al.: Optimal control theory and parameter optimization. M. National Defence. Industrial Press, New York (2005)
8. Bertsekas, D.P., Homer, M.L., Logan, D.A.: Missile defense and interceptor allocation by neuro-dynamic programming. J. IEEE Trans. Syst., Man, Cybern. A 30(1), 42–51 (2000)
9. Werbos, P.J.: Advanced forecasting methods for global crisis warning and models of intelligence.J. Gen. Syst. Yearbk. 22, 25–38 (1977)
10. Wang, F., Zhang, H., Liu, D.: Adaptive Dynamic Programming. J. IEEE Computational Intelligence Magazine, 39–47 (2009)

Power System Load Forecasting Based on EEMD and ANN

Wanlu Sun, Zhigang Liu, and Wenfan Li

School of Electrical Engineering, Southwest Jiaotong University,
Chengdu 610031, China
sunwlchina@163.com

Abstract. In order to fully mine the characteristics of load data and improve the accuracy of power system load forecasting, a load forecasting model based on Ensemble Empirical Mode Decomposition (EEMD) and Artificial Neural Networks (ANN) is proposed in this paper. Firstly, the load data can be resolved into a limited number of Intrinsic Mode Function (IMF) components and one remainder by EEMD which avoids the mode mixing problem of Empirical Mode Decomposition (EMD). Then, through the observation of the spectrum by Hilbert transform, it's obvious that the regularity and periodicity of low frequency components are stronger than high frequency components. So one sole appropriate ANN forecasting model is chosen for each low frequency component, and the linear combination of ANN model is applied to forecasting each high frequency component. Simulation results show that the new model proposed in paper is better than anyone ANN forecasting model.

Keywords: EEMD; load forecasting; one sole ANN model; linear combination of ANN model.

1 Introduction

Load forecasting is important for the development and operation of power system. For the short-term load forecasting, raising the accuracy of forecasting can both improve the economical efficiency and enhance the reliability of power system operation[1]. Many scholars have made a lot of extensive research, and proposed a variety of forecasting methods, such as Time Series Method, AR Model Algorithm, Fuzzy Algorithm, Expert System, Wavelet and ANN Algorithm. In recent years, a new method of load forecasting based on Empirical Mode Decomposition (EMD)[2], has been proposed. The complex power system load data can be decomposed into several Intrinsic Mode Function (IMF) components and a remainder by EMD, so that the feature and the regularity of the load are mined completely. And the forecasting object of EMD changes from the complex data to a series of components with certain characteristics. The accuracy of the forecasting result is greatly improved. But sometimes there is a problem called mode mixing existing in the process of EMD. It makes the IMFs lose the original physical meaning and weaken the regularity. Thereby, the forecasting accuracy may be reduced. A new method[3] called Ensemble Empirical

© Springer-Verlag Berlin Heidelberg 2011

Mode Decomposition (EEMD) which can effectively improve the mode mixing of EMD is introduced. A forecasting model based on EEMD and ANN is proposed in this paper. Firstly, the power system historical load data is decomposed into several mono-component signals by EEMD. Secondly, through the characteristics and the influence factors of the components, the appropriate ANN model is chosen for each decomposed IMF to forecast. Lastly, we take the superposition of each component forecasting results as the ultimate forecasting value. The result of load forecasting indicates that this proposed method can achieve a higher accuracy.

2 EMD and EEMD

2.1 EMD Theory

EMD[4] is a process to decompose the time series signal into many different kinds of time series components. Any complex data can be divided into a limited number of IMFs by it. Each local oscillation structure or frequency structure of signal can be described by the IMFs. EMD method is adaptive and efficient. This decomposition is based on local time scale, so that EMD is very suitable to analyze the non-linear and non-stationary signal. The steps to extract IMF from a signal $s(t)$ are as follows:

(1) Make $x(t)$ equal the value of original signal $s(t)$.

(2) Spline interpolates between local maxima (minima) of $x(t)$ to obtain the upper (lower) envelope, and calculate the mean— $m(t)$ of the upper and lower envelope.

(3) Subtract $m(t)$ from $x(t)$, get the result— $h(t)$.

(4) If $h(t)$ meets the two properties, it's an IMF. If not, $h(t)$ as a new $x(t)$ repeats the 2, 3, 4 steps.

(5) Subtract the IMF from original to obtain the residue. Use the residue to extract the new IMF, until the residue becomes a monotonic function. At last, the original is represented as:

$$s(t) = \sum_{i=1}^{n} c_i + r_n \tag{1}$$

r_n is the residue of data $s(t)$, c_i is the ith IMF obtained from $s(t)$.

However, sometimes EMD can not decompose the original data sequence correctly. The IMF, sifted out from the original signal, is not a mono-component. Some signals with different scales exist in the same IMF, or the signals with a similar scale exist in different IMFs. These IMFs extracted by EMD lose physical meaning and weaken the regularity. This is so called mode mixing.

2.2 EEMD Theory

In order to improve the mode mixing of EMD, a new method called EEMD[5] was proposed in 2005. The process of EEMD is as follow:

(1) Add a white noise to the targeted data.
(2) Decompose the data with added white noise into IMFs.
(3) Repeat step 1 and step 2 again and again, but with different white noise series each time.
(4) Obtain the (ensemble) means of corresponding IMFs of the decompositions as the final result.

$$\overline{C_j}(t) = \frac{1}{N}\sum_{n=1}^{N} C_{jn}(t) \tag{2}$$

$\overline{C_j}(t)$ is the *j*th IMF decomposed from the original signal by EEMD, and N is the number of added white noise.

The essence of EEMD eliminating mode mixing is that the uniform distribution statistical characteristic of minimal amplitude white noise is used. With the added white noise, the signal will be continuous in different scales, and different scales of the signal region will be automatically corresponded to the suitable scale. So the problem of mode mixing is avoided by EEMD[6].

3 Load Forecasting Model Based on EEMD and ANN

3.1 Overview for Neural Network

Artificial Neural Networks (ANN)[7] is one of the most popular forecasting techniques. With the strong nonlinear mapping ability, ANN can obtain the inherent law of known data easily. In this paper, the linear neural network, BP (Back propagation) network and RBF (Radial Basis Function) network are used. A brief introduction of each model is made below.

Linear neural network, made up with one or more linear neuron, is one of the simplest networks. The transfer function of each neuron is a linear function, so output of linear neural network can be any value. BP network is the multilayer feed forward neural network based on back propagation algorithm. The problem of sample between input and output changes into the problem of nonlinear optimization. Compared with other types of ANN, a novel neural network—RBF network is stronger in physiological basis, simpler in structure, faster in learning and more excellent in approaching performance.

3.2 Forecasting Model

Load forecasting based on ANN is one of the most appropriate methods in power system applications. It's also an issue been extensively studied now. There is no need to determine the relationship between the input and output in the neural network. It's just according to train the historical data, to obtain the relationship between the forecast results and the variables such as weather. Using this relation, it's easy to predict the future load. Load series of power system is non-stationary time series with certain periodicities and random. So it can be regarded as a linear combination of sub-serials characterized by different frequency. Each component shows the similar periodic changes with similar frequency characteristics and changes of the same. If electric power load forecasting is only operated by ANN, the result is to deviate from the truth. And its reliability is to reduce with the influence of random facts[8].

For the forecasting model's choice, the paper applies a method which is the suitable combination of several ANN forecasting models. The basic idea is that the actual value has the linear relationship with the forecasting value of the several networks. It needs to consider this several forecasting results totally and gets the weight coefficient through the mathematical method. The forecasting combination model is more systematic and comprehensive than a model. The linear combination method[8-9] is :

$$\begin{cases} y_t = k_1 \hat{y}_{1t} + k_2 \hat{y}_{2t} + k_3 \hat{y}_{3t} \\ k_1 + k_2 + k_3 = 1, k_i \geq 0 (i = 1,2,3) \end{cases} \tag{3}$$

Where $\hat{y}_{1t}$, $\hat{y}_{2t}$, $\hat{y}_{3t}$ are the forecasted load value from BP networks, RBF networks, linear networks, and k1, k2, k3 are the weigh coefficients of each one, y_t is the combined predictive value.

The steps forecasting model proposed in the paper are shown in Fig. 1[10].

(1)Using EEMD, load data can be decomposed into several IMFs which are IMF1, IMF2, …, IMFn and res. The spectrum of each IMF is arranged by frequency.
(2)In order to select the suitable forecasting model, the spectrum of each IMF is calculated with Hilbert transform.
(3)The superposition of each forecasting results are obtained as the ultimate forecasting value.

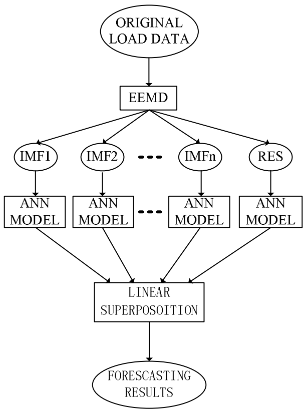


Fig. 1. Hybrid forecasting model of EEMD and ANN

4 Simulation of Load Forecasting

The power system load data, which is used to forecast the next day's load data (24 points), comes from a certain place of Sichuan Province from June 30 to August 5 in 2006. With the help of EEMD, the load sequence can be resolved into a series of independent IMF components with different scales. Fig. 2 shows the waveforms of load data and the sifted IMF components. The instantaneous frequency of each component is shown in Fig. 3.

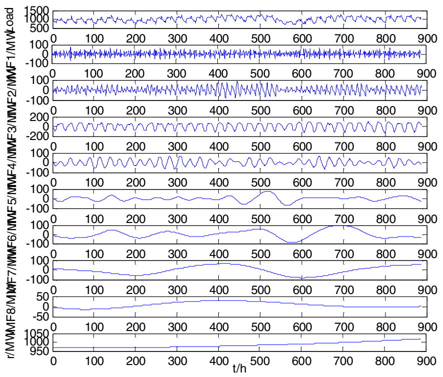


Fig. 2. Original load and the sifted components of EMD

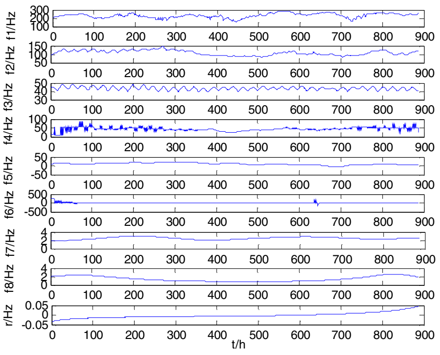
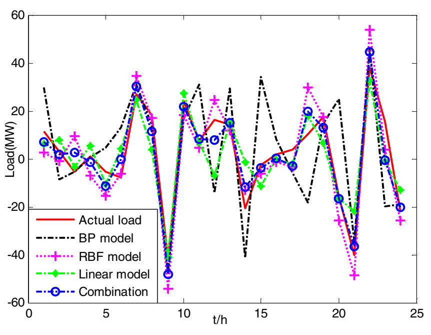


Fig. 3. Spectrum of each IMF

From the Fig. 3, it's obvious that each of the first two IMFs is high frequency component, and has a wide band and strong fluctuations. Simulation results show that it's hard to find a suitable one sole ANN forecasting model for this kind of component to achieve high forecasting precision. So a liner combination model is applied in the forecasting for IMF1 and IMF2. Weight coefficients can be calculated by the formula in [9].



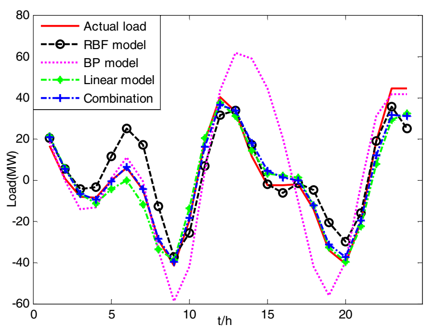


Fig. 4. Each model forecasting curve of IMF1 **Fig. 5.** Each model forecasting curve of IMF2

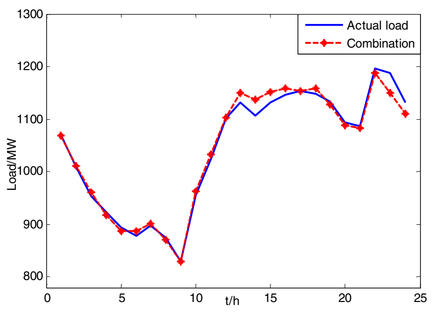
Table 1. Comparison of forecasting error of IMF1

model	BP	RBF	Linear	Combination
Maximum difference	35.097	19.507	23.476	15.806
Mean difference	14.456	7.419	7.615	4.985

Table 2. Comparison of forecasting error of IMF2

model	BP	RBF	Linear	Combination
Maximum difference	13.303	3.507	23.476	15.806
Mean difference	13.608	8.084	6.716	3.436

Through the figures and tables above, the forecasting result of liner combination model is more accurate than the results of BP model and RBF model. Each frequency band of IMF3-IMF6 is narrow and these components are basically stable. According to the characteristics of the waveform and frequency, the forecasting models of rest components are as follows: RBF model for IMF3 and IMF4, BP model for IMF5 and IMF6. Because the frequencies of IMF7, IMF8 and the remainder are low, and the volatility of them is weak, liner model is selected for them. The ultimate forecasting value that is the superposition of every forecasting result is shown in Fig. 6.



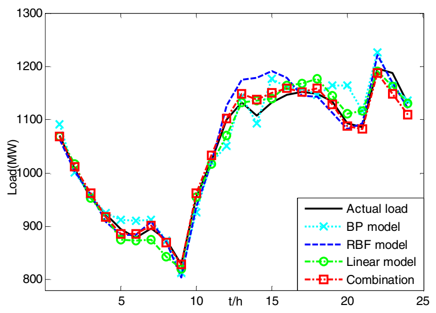


Fig. 6. Forecasting curve of Aug.6 with the combination model

Fig. 7. Forecasting curve of Aug.6 with each model

In order to verify the high precision and accuracy of proposed combination model based on EEMD and ANN, the result forecasted only by BP model or RBF is compared with result forecasted by proposed model. Fig. 7 shows the curves forecasted by each model. Table 3 lists the relative error and mean error of each forecasting model. Though the Fig. 7, it is clear that the combination method achieves a better forecasting result than other methods. From the Table 3, the relative error and mean error forecasted by the model proposed in paper are smaller than those forecasted by RBF model and BP model directly. The validity of the proposed method is verified.

Table 3. Comparison of the 24-hour forecasting error of Aug.6

Time (hour)	BP Error (%)	RBF Error (%)	Combination Error (%)	Time (hour)	BP Error (%)	RBF Error (%)	Combination Error (%)
0	2.036	0.660	0.055	12	1.040	3.821	1.470
1	0.734	0.565	0.248	13	1.194	6.529	2.795
2	0.262	0.789	0.826	14	3.947	5.222	1.695
3	0.263	1.584	0.566	15	1.296	2.815	1.037
4	1.974	1.215	0.896	16	0.628	0.590	0.031
5	3.706	0.675	0.839	17	0.114	0.359	1.003
6	1.739	1.408	0.483	18	2.637	1.819	0.466
7	0.120	0.426	0.481	19	6.531	0.928	0.362
8	1.910	2.909	0.050	20	2.788	0.618	0.361
9	3.038	1.983	0.764	21	2.392	2.066	0.808
10	0.177	0.480	0.996	22	1.879	1.438	3.246
11	4.420	2.469	0.175	23	0.421	0.826	1.901
Maximum error					6.531	6. 529	3.245
Mean error					1.885	1.758	0.898

5 Conclusions

In this paper, a new load forecasting model based on EEMD and ANN is proposed. With EEMD, the load data sequence can be divided into a limited number of IMF components and remainder. Through the spectrum analyze with Hilbert transform, a suitable forecasting model can be chosen for each component. Finally, the superposition of every forecasting result is taken as the ultimate forecasting value. The forecasting relative error of the model proposed in this paper is compared with the relative error of BP neural network model and error of RBF neural network model. The result indicates that new method is effective to achieve the higher forecasting accuracy.

Acknowledgment

The authors would like to thank National Nature Science Foundation of China (51007074), Program for New Century Excellent Talents in University (NECT-08-0825), Fok Ying Tung Education Fund (101060), Sichuan Province Distinguished Scholars Fund (07ZQ026-012) and Fundamental Research Funds for the Central Universities (SWJTU09ZT10) in China.

References

1. Hongxiao, W.: Load Forecasting Based on Soft Computing, pp. 4–13. Shanghai Jiao Tong University, Shanghai (2007)
2. Zhihui, Z., Yunlian, S., Yu, J.: Short term Load Forecasting Based on EMD and SVM. High Voltage Engineering 33(5), 118–122 (2007)

3. Wu, Z., Huang, N.E.: Ensemble empirical mode decomposition: a noise-assisted data analysis method. Calverton: Centre for Ocean-Land-Atmosphere Studies, Technical Report 51(193) (2005)
4. Weili, B., Zhigang, L., Qi, W., Dengdeng, Z.: Load Forecasting of Power System Based on HHT. Sichuan Electric Power Technology 32(3), 9–13 (2009)
5. Dai, L., Songling, P., Wei, L.: Power System Short-Term Load Forecasting Based on EEMD and Dynamic Neural Network. Journal of Northeast Dianli University(Natural Science Edition) 29(6), 20–26 (2009)
6. Jinshan, L.: Gearbox Fault Diagnosis based on EEMD and Hilbert Transform. Journal of Mechanical Transmission 34(05), 62–65 (2010)
7. Ying, W., Dingfan, C., Xiaobing, T.: Sum-marizing of Neural Network. Science & Technology Progress and Policy 6, 133–134 (2002)
8. Bai, W., Liu, Z., Zhou, D.: Research of the Load Forecasting Model Base on HHT and Combination of ANN. In: Asia-Pacific Power and Energy Engineering Conference (2009)
9. Linchuan, L., Dong, L., Wenjie, W.: A linear combination based simplified load forecasting method for power system. Power System Technology 26(10), 10–13 (2002)
10. Weili, B., Zhigang, L., Quanwei, P., Jian, X.: Research of the load forecasting model based on HHT and combination of ANN. Power System Protection and Control 37(19), 31–35 (2009)

Analog Circuit Fault Diagnosis with Echo State Networks Based on Corresponding Clusters

Xiyuan Peng, Jia Guo, Miao Lei, and Yu Peng

Automatic Test and Control Institute, Harbin Institute of Technology,
Harbin 150080, China
guojia.gm@gmail.com

Abstract. Analog circuit fault diagnosis can be modeled as a pattern recognition problem. Fault patterns are complicated which has high demands for classification accuracy and efficiency. Therefore a new analog circuit fault diagnosis method using Echo State Networks (ESNs) is proposed. We adopt the time windows function to construct reservoir with corresponding clusters of ESNs inspired by complex network topologies imitating cortical networks of the mammalian brain. Multiple-cluster reservoir is generated instead of non-clustering reservoir of the original ESNs with random sparse connections. We use the number of classes to determine the number of clusters to improve performances in specific analog circuit fault diagnosis problems. Simulation results show the effectiveness of the proposed method.

Keywords: Echo State Networks; Time windows; Analog circuit fault diagnosis.

1 Introduction

Analog circuits are the most unreliable part to malfunction in electronic equipments. Fault diagnosis of analog circuits can improve the maintenance of electronic equipments. Due to the component tolerances and nonlinear effects, complex relationships exist between circuit response and component parameter in analog circuit [1]. Researchers introduced neural networks into analog circuit fault diagnosis [2]. Thus, fault diagnosis is modeled as pattern recognition problem. However, traditional neural networks have several defects such as high computational training costs, which cannot meet the needs of practical fault diagnosis.

In 2001, H. Jaeger presented Echo State Networks (ESNs) [3] which provided an architecture and learning principle without the slow convergence for Recurrent Neural Networks (RNNs). By mapping the input to a higher dimension by the dynamics of a random, fixed dynamical system called dynamical reservoir (DR) and only training a simple output mechanism, ESNs has achieved state-of-the-art performance in practical problems, such as time series prediction [4], classification [5] and anomaly detection [6]. Traditionally, people used ESNs for time domain tasks. ESNs for static pattern classification presented in [7] achieve good performances. However, it is difficult to use DR without clusters to meet the needs of classification problems of different categories of data.

© Springer-Verlag Berlin Heidelberg 2011

In real-world complex networks, researchers have discovered small-world phenomena and scale-free properties in the past few years. Recently, several new neural network models that have some characteristics of complex network topology attracted the attention of researchers. Associative ESNs with either small-world architecture or scale-free topology have been investigated, and most of which demonstrate better performance than the randomly connected.

Kaiser et al. proposed a multi-cluster cortical network generation method [8]. The generated networks simulate certain characteristics of mammalian cerebral cortices well. Two growth mechanisms manage the generating process. One depends on the spatial-distance between neurons while the other depends on specific time-windows.

In order to generate a clustered classifier and improve the accuracy of analog circuit fault diagnosis, complex network theory is introduced to generation of ESNs dynamic reservoir. Moreover, to meet the demands of specific classification tasks, we associate the numbers of clusters in dynamic reservoir with the number of classes.

The remainder of the paper is structured as follows. In Section 2 we introduce analog circuit fault diagnosis. Section 3 presents echo state networks based on corresponding clusters. We present the experiment results and discussion in Section 4 followed by the conclusions in Section 5.

2 Introduction of Analog Circuit Fault Diagnosis

Traditional approaches, such as fault dictionary technique and parameter identification method, have several difficulties associated with analog fault diagnosis. For instance, fault dictionary technique always deal with the single fault and hard fault situation due to the component tolerances and the limited volume of fault dictionary. And we can hardly identify all the circuit parameters due to the large number of elements in analog circuit. Neural networks overcome these difficulties.

Analog circuit fault diagnosis with neural networks is built up with two phases: learning and diagnosis. In the learning phase, the responses of circuit under test (CUT) were obtained by intentionally introducing faults to generate fault data samples. The fault-free components are varied within their tolerance ranges. Thus, the circuit responses reflect the faults status. A diagnosis model with neural networks can be buit up with the fault data samples and the fault classes. In addition, feature extraction is always adopted to improve the diagnosis performance, because the data samples are always high-dimentional due to the small sampling interval. In the diag nosis phase, the diagnosis model is fed with testing sets for fault classification. The generalizability of the model can be evaluated during this phase.

Fault patterns are complicated which has high demands for classification accuracy and efficiency. Therefore, we propose a new analog circuit fault diagnosis method using ESNs as fault diagnosis model.

3 ESNs Based on Corresponding Clusters

3.1 Introduction of Echo State Networks

ESNs maps the input to a higher dimension by a random, fixed dynamical architecture called reservoir. The basic idea of ESNs is shared with Liquid State Machines (LSM). Both are subsumed under the name of Reservoir Computing.

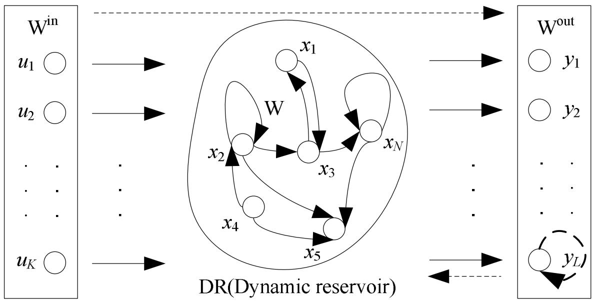


Fig. 1. Scheme diagram of a typical structure of ESNs

Fig. 1 shows a typical structure of ESNs which consists of input units, dynamical reservoir (DR) and output units. ESNs with K input units, N dynamical reservoir processing elements and L output units can be described as

$$x(n+1) = f(W^{in}u(n+1)+Wx(n)+W^{back}y(n)) \tag{1}$$

$$y(n+1) = f^{out}(W^{out}(u(n+1),x(n+1),y(n))) \tag{2}$$

In (1) and (2), $x(n) = (x_1(n),\ldots, x_N(n))$, $y(n) = (y_1(n),\ldots, y_L(n))$, $u(n) = (u_1(n),\ldots, u_K(n))$ are activations of the DR processing elements, output units and input units at time step n, respectively. The functions $f = (f_1,\ldots, f_N)$ are activation functions for DR processing elements. The functions $f^{out} = (f_1^{out},\ldots,f_L^{out})$ are the output units' output functions (implemented as identity functions in this paper). By an $N\times K$ input weight matrix $W^{in} = (w_{ij}^{in})$, the input is tied to DR processing element. The DR processing elements are connected by an $N\times N$ matrix $W = (w_{ij})$. $W^{back} = (w_{ij}^{back})$ is an $N\times L$ matrix for the connections that project back from the output to DR. And DR is tied to the output units by an $L\times(K+N+L)$ matrix $W^{out} = (w_{ij}^{out})$. The term $(u(n + 1), x(n + 1), y(n))$ is the concatenation of the input, internal, and previous output activation vectors.

ESNs learning process can be briefly described as training suitable output weights to obtain the desired output from DR which is random and fixed in advance. Determination of optimal output weight matrix W^{out} is a linear regression task of mean-square error (MSE) minimization [3][9].

ESNs have been used traditionally for time domain tasks. The way to use ESNs for static pattern classification was firstly presented by Embrechts and Alexandre [7]. The issue was to break the dependency of activations of DR processing elements. Equation (3) shows the stabilization phase of x(n). Every input sample is held until x(n) does not change significantly. The index n in Equation (3) is only used to distinguish between different patterns and does not represent time.

$$x(n+1)^{(i)} = W^{in}u(n+1)+Wx(n+1)^{(i-1)} \tag{3}$$

Before using the activation function, x(n) has been stabilized. The method keeps the advantages of ESNs and achieves good performances. However, the reservoir is generated randomly without using priori knowledge of the number of classes. It is difficult for the connection structure without clusters to meet the needs of classification of different data categories.

3.2 The Proposed Method

In order to meet the specific classification demands and improve the classification accuracy, complex network theory was introduced to the dynamic reservoir of the original ESNs. Note that one of effective mechanisms, multi-cluster cortical networks by time windows, was carried out by Kaiser.

In Kaiser's research, it shows that many neural networks, such as the complex cortical networks of the mammalian brain [10], are organized in multiple clusters, with many connections within but few links between clusters. Motivated by this discovery, he introduced this time window mechanisms. Reference [8] took advantage of this time window mechanism to generate clustered dynamic reservoir of ESNs, combined with complex network theory.

Based on the above results and discoveries, this paper proposes ESNs based on corresponding clusters for static classification. We expect this structure to generate the relationship between the number of clusters in our ESNs and the number of classes in different tasks.

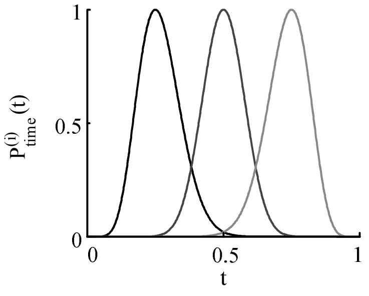


Fig. 2. Plot of $P^{(i)}_{time}$ (k=3, α=0.2)

In our method, connection of two nodes is established with probability

$$P = P_{dist}(d(U,V)) \cdot P^{(w(V))}_{time}(t) \cdot P^{(w(U))}_{time}(t) \tag{4}$$

where $P_{dist}(d(U,V)) = \beta e^{-\gamma \cdot d(U,V)}$ is the distance-dependent probability. $d(U, V)$ denotes the Euclidean distance between thenodes U and V.

$P^{(i)}_{time}(t) = P(t, \mu^{(i)}, \sigma(\alpha)), (i = 1,\ldots,k)$ is the time window dependent probability. α is the time window integral value. k is the number of pioneer nodes as well as the number of clusters which is determined according to the number of classes in this paper. Fig. 2 shows the time window function functions with k=3 and α=0.2. Fig. 3 shows the clusters in the reservoir of ESNs corresponding to the number of classes with specific data (k=3). Due to space constraints, we will not go into details of the parameter settings, and refer the reader to [8].

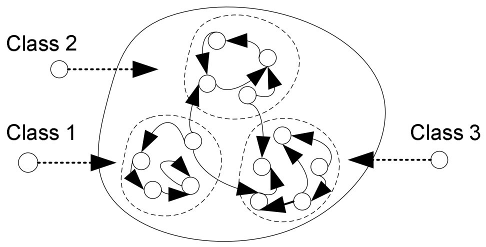


Fig. 3. Scheme diagram of DR with corresponding clusters (*k*=3)

In order to generate the connections of reservoir processing elements, we implement the method by the following five steps.

1) Set the parameters of the algorithm including *N*, α , β , γ representing the size of reservoir, the time window integral value, the distance-dependent parameters. We set the number of pioneer nodes - *k* according to the number of classes.

2) Pioneer nodes are deployed in a two-dimensional plane with range [0,1] randomly. Store their spatial coordinates. In this paper, we don't use a highly symmetric placement of the pioneer nodes which is used in [8] for brain simulation [11].

3) New node *U* is placed randomly and corresponds to the nearest pioneer node. Then compute the time window dependent probability $P_{time}^{(w(U))}(t)$.

4) The distance-dependent probability $P_{dist}(d(U,V))$ and the time window dependent probability $P_{time}^{(w(V))}(t)$ of new node *U* and existing nodes *V* are computed. Then the connection probabilitys *P* are calculated by Equation (4). Connections exist if the connection probabilities are not smaller than some random number.

5) Repeat 4) until all the connection probabilities of new node *U* are computed and compared. If the connections of new node *U* and existing nodes *V* do not exist, then go back to 3) until the number of existing nodes is *N*.

3.3 Training of ESNs

In this paper, the activation functions for DR processing elements were implemented as tanh functions and the output units' output functions f^{out} were implemented as identity functions. The connections denoted as dashed line in Figure 1 were not adopted. We state the training of ESNs in this paper as: 1) Set the parameters of ESNs. 2) Initialize connection matrices W^{in} and generate W by our method presented in Section 3.1. 3) Collect x(n) by feeding training samples into Equation (3). 4) Calculate W^{out} with pseudo-inverse method by Equation (2).

4 Experiments and Discussions

In this paper, analog circuit fault diagnosis experiments are built up with two phases: learning and diagnosis. In the learning phase, the impulse responses of circuit were obtained and processed by Haar wavelet transform. Then we selected the first coefficient of approximation levels 1–5 as features fed into ESNs for training the diagnosis model. Note that, feature extraction is not the focus of this paper. Moreover, feature selection can be beneficial to improve the results of the method which will be addressed by future work. In the diagnosis phase, testing sets were obtained by the same procedure and fed it to the diagnosis model for fault classification.

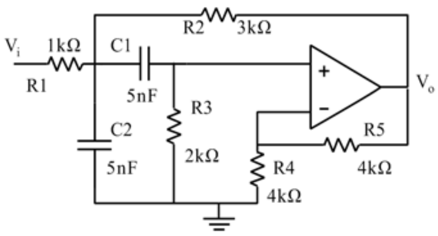


Fig. 4. Sallen-Key bandpass filter (S-K)

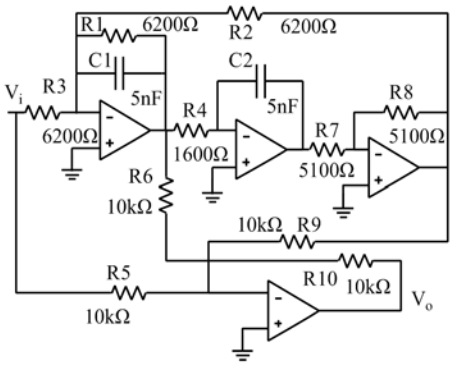


Fig. 5. Four op-amp biquad high-pass filter (4-OPA)

We adopted a Sallen-Key bandpass filter circuit (Fig. 4) and a four op-amp biquad high-pass filter circuit (Fig. 5) using PSPICE software in experiments. In order to simulate working condition for circuit more precisely, the resistors and capacitors are assumed to have tolerances of 5% and 10%, respectively. We conduct 50 times of Monte Carlo analysis to acquire fault samples by varying components within their tolerances to every kind of fault class. We get 8 and 12 kinds of fault classes from S-K circuit and 4-OPA circuit, respectively. Due to space constraints, we will not go into details of the faults setting, and refer the reader to [2].

For each dataset, we did 10 repetitions of a 2-fold cross validation and adopted same parameters including the number of DR processing elements, ESNs reservoir spectral radius and DR connectivity for generating reservoir of the original ESNs and our method.

We also obtained results from experiments with the original ESNs and MLPs with BP training (BPNN) as fault diagnosis methods. We obtained the time consuming data in seconds by calculating the average of 100 continuous operations with 90% of samples as training sets and 10% as testing sets.

Table 3 shows the diagnosis results. Our method achieves the lowest error rate. We proved the advantages of reservoir with corresponding clusters by the better performance than the original ESNs. The training time of our method is significantly lower than BPNN and also lower than original ESNs. Due to the additional time consumed by reservoir generation with corresponding clusters, the time for generating DR of our method is higher than original ESNs. Note that generation of reservoir can be completed independently before training. Therefore, the time for generating DR has little effect on the applications.

Table 3. Results comparisions

Circuit	Method	BPNN	Original ESNs	Our method
S-K	Error rate	10.11%	5.33%	4.91%
	Training time	3.309	0.162	0.139
	Tesing time	0.009	0.017	0.014
	Generating time	——	0.071	0.926
4-OPA	Error rate	17.88%	8.91%	8.37%
	Training time	8.031	0.197	0.117
	Tesing time	0.010	0.017	0.010
	Generating time	——	0.134	0.717

Experiment results indicate that the methods which adopt ESNs outperform the BPNN in analog circuit fault diagnosis. Reservoir with corresponding clusters is beneficial for improving the diagnostic performance with priori knowledge of the number of fault classes. Our method achieves good performance on time consumed, and can meet the needs of analog circuit fault diagnosis.

5 Conclusions

Motivated by the discovery that, the complex cortical networks of the mammalian brain are organized in multiple clusters, we propose an interesting model——ESNs with corresponding clusters for analog circuit fault diagnosis. Experiment results show that the new method outperforms the original ESNs and BPNN.

Acknowledgments. This work is supported in part by a Grant (No. 20092302110013) from Ph.D. Programs Foundation of Ministry of Education of China and a Grant (NCET-10-0062) from Program for New Century Excellent Talents in University, Ministry of Education of China.

References

1. Liu, R.W.: Testing and diagnosis of analog circuits and systems. Van Nostrand Reinhold, NY (1991)
2. Aminian, F., Aminian, M., Collins, H.W.: Analog Fault Diagnosis of Actual Circuits Using Neural Networks. IEEE Transactions on Instrumentation and Measurement 51(3), 544–550 (2002)

3. Jaeger, H.: The “echo state” approach to analysing and training recurrent neural networks. German National Research Center for Information Technology, Fraunhofer Institute for Autonomous Intelligent Systems, Tech. Rep., GMD Report 148 (2001)
4. Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic system and saving energy in wireless communication. Science 304, 78–80 (2004)
5. Skowronski, M.D., Harris, J.G.: Automatic speech recognition using a predictive echo state network classifier. Neural Networks 20, 414–423 (2007)
6. Ding, H.-Y., Pei, W., He, Z.-Y.: A Multiple Objective Optimization Based Echo State Network Tree And Application To Intrusion Detection. In: IEEE Int. Workshop VLSl Design & Video Tech., pp. 443–446 (2005)
7. Embrechts, M., Alexandre, L., Linton, J.: Reservoir computing for static pattern recognition. In: 17th European Symposium on Artificial Neural Networks, Bruges, Belgium (2009)
8. Nisbach, F., Kaiser, M.: Developmental time windows for spatial growth generate multiple-cluster small-world networks. The European Physical Journal B 58, 185–191 (2007)
9. Jaeger, H.: Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the ”echo state network” approach. German National Research Center for Information Technology, Fraunhofer Institute for Autonomous Intelligent Systems, Tech. Rep., GMD Report 159 (2002)
10. Kaiser, M., Hilgetag, C.C.: Development of multi-cluster cortical networks by time windows for spatial growth. Neurocomputing 70, 1829–1832 (2007)
11. Song, Q., Feng, Z.: Effects of connectivity structure of complex echo state network on its prediction performance for nonlinear time series. Neurocomputing 73, 2177–2185 (2010)

Anti Boundary Effect Wavelet Decomposition Echo State Networks

Jianmin Wang, Yu Peng, and Xiyuan Peng

Automatic Test and Control Institute, Harbin Institute of Technology,
Harbin, 150080, China
wangjianmin@hrbust.edu.cn,
{pengyu,pxy}@hit.edu.cn

Abstract. We propose a novel approach based on wavelet decomposition and echo state networks to discover the multiscale dynamics of time series which we call anti-boundary-effect wavelet decomposition and echo state networks (ABE-WESNs). ABE-WESNs use the wavelet decomposition as preprocessing steps and choose a matched ESNs for every scale level. We use the data extension methods to overcome the boundary effect. The introduced weight factors can both resolve the problem of cumulation of errors resulting from the wavelet decomposition. Experiments and engineering applications show that the ABE-WESNs can accurately model and predict some time series with multiscale properties.

Keywords: Echo State Network, Wavelet, Time Series Prediction, Boundary Effect.

1 Introduction

As a new paradigm in recurrent neural network training, Echo State Networks (ESNs) have recently attracted great interest because of its simple mathematical formulation, simple and linear learning algorithm and superior performance compared with other classical methods. It has become a vivid research field with numerous extensions of the basic idea and been successfully used to a variety of applications, such as time series predictions, pattern classification, object tracking and motion prediction. However, ESNs is still in its infancy and needs further improvements and extensions, such as reservoir design and adaptation methods[1].

One of the shortcomings of classical ESNs is that even though the reservoir is sparse, the activations are still coupled so strongly that the ESNs is poor in dealing with different time scales simultaneously, e.g., predicting several superimposed generators [1]. To model and predict multiscale time series with ESNs, there are two different ideas. One is to make some modifications in ESNs itself, such as changing the types of neuron activation functions. By replacing part of the neuron functions in the reservoir with sigmoid-wavelet functions, Se Wang et al proposed a novel model called sigmoid-wavelet hybrid echo state networks (SWHESN) to improve the performance of classical ESNs when it is used

© Springer-Verlag Berlin Heidelberg 2011

as chaotic predictor[2]. By dividing the reservoir into decoupled subreservoirs and introducing inhibitory connections among all the sub-reservoirs, Yanbo Xue proposed a new structure of reservoir called decoupled echo state networks [3]. Simulation experiments show that the decoupled echo state networks can tackle some multi-scale time series prediction problem. Designing a finite impulse response or infinite impulse response filter as neurons in reservoirs can also enable ESNs to simultaneously learn multiple attractors and signals at different time scales, which is especially important for modeling real-world time series[4,5]. It is natural to design hierarchical models to solve multi-timescale and/or spatial scale problem. The authors of [6] proposed hierarchical echo state networks to discover multiscale dynamical features of dynamical system. These methods of process multiscale problem mentioned above have a common feature that the training process of model is too complex to be applied to practical applications.

Another way to solve multiscale problems is straightforward. It introduces the wavelet decomposition in the model as the preprocessing step. Francis Wyffels et al. took wavelet decomposing as preprocessing step to capture the information at different scales and determine a matched ESNs for different scale levels. We call this model wavelet decomposition echo state networks(WESNs)[7]. Although WESNs can model and predict all the three time series defined in ESTSP 2008, there are still some problems. First, there is no consideration on boundary effect of wavelet decomposition which actually exists in wavelet decomposition [11]. Second, there may be the cumulation of errors generated by different ESNs. The original WESNs model[7] determined a matched ESNs model for each scale level. For every ESN, people carry out the training for minimizing the root mean square error of this scale level. Thus, the model can only guarantee the accuracy of every scale level instead of the whole time series. In other words, it may undergo a large cumulation of errors.

As a result, we propose a enhanced model to overcome the problems, which we call anti-boundary-effect wavelet decomposition Echo State Networks (ABE-WESNs). Compared with the original WESNs, there are two major modifications. First, to overcome the boundary effect resulting from wavelet decomposing, we truncate the data without boundary distortion and use the truncated data as the input of every matched ESNs of a different time scale. Second, we introduce the weight coefficients, also known as the weight factors here, to the ABE-WESN model to solve the problem of error cumulation. Specifically, we consider the final output of the model as a linear combination of the outputs of the matched ESNs other than a summation of them. The introduction of weight factor can also make the parameter determination of the matched ESNs of every time scale easy. We can obtain the weight factors by minimizing errors of the output of the ABE-WESNs model and teacher samples.

The organization of this paper is as follows. In section 2, we describe the structures of the ABE-WESNs in detail. We also discuss the structure of the models and training algorithm of the parameters. We describe some experiments and real-life world engineering applications with the proposed approach in detail in section 3. We conclude this paper in section 4.

2 ABE-WESNs Model

2.1 The Structure of ABE-WESNs Model

The aim of the ABE-WESNs is to simultaneously model and predict in every time scale and as a whole. Therefore, we introduce the wavelet decomposition as the preprocessing step to capture useful information at different time scales and choose different ESNs to match different properties of every time scale, which may result in the ESNs having totally different parameters. To improve the overall predicting accuracy, we introduce weight factors. Fig. 1 shows the structure of the ABE-ESNs in detail. According to Fig. 1, the ABE-WESNs consist of four submodules: wavelet decomposition and single-branch reconstruction, boundary-effect processing, ESN models and weight factor.

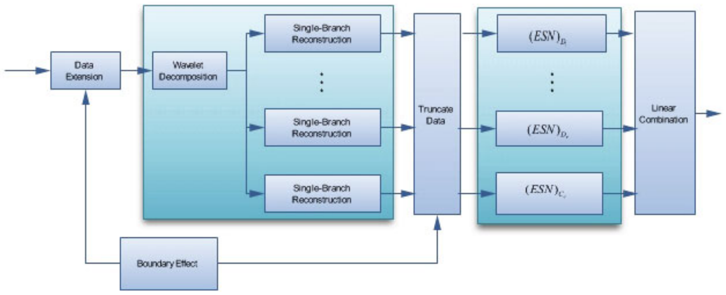


Fig. 1. The Structure of the ABE-WESNs

2.2 Wavelet Decomposition and Single-Branch Reconstruction

Let $\{c_0^{m_0}, m \in \mathbb{Z}\}$ denote the multiscale time series, where m is the length of the sequence. We describe the wavelet decomposition as follows.

$$\begin{cases} c_j^{m_j} = \sum\limits_{m_{j-1} \in \mathbb{Z}} h_{m_{j-1}-2m_j} c_{j-1}^{m_{j-1}} \\ d_j^{m_j} = \sum\limits_{m_{j-1} \in \mathbb{Z}} g_{m_{j-1}-2m_j} c_{j-1}^{m_{j-1}} \end{cases} \tag{1}$$

In (1), $d_{j-1}^{m_{j-1}}$ are the detail coefficients, which mainly consist of a high-frequency noise; $c_{j-1}^{m_{j-1}}$ are the approximation coefficients, which contain much less noise than that of the original signal. $j-1$ corresponds to the number of levels. $m_j - 1$ is the length of the series. h and g are the high-pass and low-pass filter respectively. The reconstruction is the inverse process of decomposition, which we can implement by (3)

$$\begin{cases} D_j = \sum_{m_1 \in \mathbb{Z}} \overline{h}_{m_1 - 2m_0} \sum_{m_2 \in \mathbb{Z}} \overline{h}_{m_2 - 2m_1} \cdots \sum_{m_j \in \mathbb{Z}} \overline{h}_{m_j - 2m_{j-1}} d_j^{m_j} \\ C_j = \sum_{m_1 \in \mathbb{Z}} \overline{g}_{m_1 - 2m_0} \sum_{m_2 \in \mathbb{Z}} \overline{g}_{m_2 - 2m_1} \cdots \sum_{m_j \in \mathbb{Z}} \overline{g}_{m_j - 2m_{j-1}} d_j^{m_j} \end{cases} \tag{2}$$

In the famous Mallat Algorithm, we decompose the original sequence $\{c_0^{m_0}\}$ level by level according to (1) and get the final high-frequency sequence $\{d_j^{m_j}, j = 1, 2, \cdots e\}$ and the low-frequency sequence $\{c_j^{m_j}, j = 1, 2, \cdots e\}$, where $m_j = (1/2)m_{j-1}, i = 2, 3, \cdots, e$ correspond to the length of the sequence $d_j^{m_j}$ or $c_j^{m_j}$. Then we can obtain the single-branch reconstruction sequence $D_j, j = 1, 2, \cdots, e$ and C_e in terms of (3), whose length are all m meeting the purpose of single branch reconstruction.

2.3 Dealing with Boundary Effect

Boundary effect, also known as Gibbs phenomenon, are common in wavelet decomposition and other methods. We summarize the mechanism of the boundary effect as follows: The discard of parts of the original sequence inevitably results in some harmonic of the spectrum being discarded or the changed on the magnitude or phase of the spectrum, which in turn result in some distortion on the boundary of the original sequence.

There are several methods to overcome the boundary effect in the wavelet decomposition, such as Zero-Value Padding, Symmetric-Value Padding, Smooth-Value Padding, algorithm of Matching Wave. In this paper, we choose the data extension method to overcome this problem. The steps are as follows.

Step 1. Truncate n samples from the original time series and construct sequence c_0^n, where $n > m_0$ and m_0 is the number of the samples we expect to use.
Step 2. Perform wavelet decomposition on the sequence c_0^n and reconstruct the sequence $D_i, i = 1, 2, \cdots, e$ and C_e, where the length of these sequence is m.
Step 3. Truncate the sequence $D_i, i = 1, 2, \cdots, e$ and C_e to make the length of these sequence are m_0.

2.4 ESN Model

We obtain $e + 1$ sequence($D_i(i = 1, 2, \cdots, e)$ and C_e) from the original sequence by decomposing the wavelet and removing the boundary effect, where $D_i(i = 1, 2, \cdots, e)$ and C_e have different properties respectively. It is difficult to predict all the sequence accurately using the same ESNs model, which can partially explain why ESN is poor in dealing with different time scales simultaneously. Therefore, it is rational to choose different ESNs for sequence $D_i(i = 1, 2, \cdots, e)$ and C_e. This means that we set up suitable parameters for every ESNs model.

(1)Choosing input and output sequences. Based on the principle of ESNs [8], we choose the following format sequence as the input of $(ESNs)_{D_j}$, where we use $(ESNs)_{D_j}$ to denote the ESN model applied to sequence D_j.

$$\begin{cases} \mathbf{u}_{D_j}(n) = [D_j(n), D_j(n-v_1), \cdots, D_j(n-v_{K-1}))] \\ \mathbf{y}_{D_j}(n) = [D_j(n+v_K)] \end{cases} \tag{3}$$

In (3), $v_i, i = 1, 2, \cdots, K$ denotes the delay time such that $v_i < v_{i+1}, i = 1, 2, \cdots, K-1$, where K is the embedding dimension. This means $\mathbf{u}_{D_j}(n)(K \times 1)$ is the input vector of time n and $\mathbf{y}(n)(1 \times 1)$ is the output of the time n. There are several methods to determine the delay time and embedded dimension, such as autocorrelation function[9], Mutual Information, C-C method. We can choose a proper method according to the characteristics of the problem.

(2)Parameters of the reservoir. The important reservoir parameters of $(ESNs)_{D_j}$ are activation function f_{D_j}, spectral radius $\rho(W)$ of the internal weight matrix W, the size of the reservoir, the scale of the input SC_{D_j}, the shift of the input SF_{D_j}, the sparse connectivity C_{D_j} of the internal weight matrix W. Based on the principle of the ESN[8], we update the states of the $(ESNs)_{D_j}$ as

$$X_{D_j} = f_{D_j}\left(W_{D_j}^{in} u_{D_j}(n) + W_{D_j} X_{D_j}(n+1) + W_{D_j}^{back} y_{D_j}(n-1)\right) \tag{4}$$

where $W_{D_j}^{in}(N_{D_j} \times K_{D_j})$, $W_{D_j}(N_{D_j} \times N_{D_j})$ and $W_{D_j}^{back}(N_{D_j} \times 1)$ are the input, internal and feedback weight matrix respectively. N_{D_j} denote the sizes of the reservoirs.

(3)Calculating Output Weight Matrix. Assuming that the initial state of the reservoir is $X_{D_j}(0) = 0$, we apply the input sequence $\{u_{D_j}(n), n = 1, 2, \cdots, T\}$ to the (4). For each time that is larger than or equal to an initial washout time T_0, we collect the corresponding network state $\{X_{D_j}(n), n = T_0, T_0{+}1, \cdots, T\}$ as a new row into a state collecting matrix M_{D_j}. In the end, one will obtain a state collecting matrix of size $(T - T_0 + 1) \times (K_{D_j} + N_{D_j} + 1)$. Similarly, for each time larger than or equal to T_0, we collect the teacher output $\mathbf{y}(n)$ row-wise into a teacher collection matrix $\mathbf{M_{D_j}}$ with the size of $(T - T_0 + 1) \times L_{D_j}$. According to the (5)

$$W_{D_j}^{out} = M_{D_j}^{-1} T_{D_j} \tag{5}$$

(4)Calculating Output of $(\boldsymbol{ESN})_{\boldsymbol{D_j}}$. According to the output equation of $(ESN)_{D_j}$, we can calculate its output as follows

$$\hat{\mathbf{y}} = W_{D_j}^{out}\left(\mathbf{u}_{D_j}(n), \mathbf{x}_{D_j}\right) \tag{6}$$

where $(\text{cot}, \cdot)$ denotes the vector concatenation. Repeating Step (1)-(4), we can obtain the output of each scale level, which means $\hat{\mathbf{y}}_{D_j}(n), j = 1, 2, \cdots, e$ and $\hat{\mathbf{y}}_{C_e}(n)$.

2.5 Weight Factor

So far, we have described how to predict the sequence of every scale level. Note that we carry out the modelling and prediction process on every scale and only use the information of this level. Therefor, we can only ensure the accuracy of each level. This inevitably results in error cumulation if the predicted results are simply summed. In this paper, we introduce weight coefficients which are also called weight factors to solve this problem. By adjusting the weight factors, we can not only improve the accuracy of the whole but also overcome the boundary effect to some extent. Therefor, the ABE-WESN can predict the multiscale time series at both different scale and the whole.

Let

$$E(n) = y(n) - \hat{y}(n) = y(n) - \left(\sum_{i=1}^{e} a_i \hat{\mathbf{y}}_{D_i} + a_{e+1} \hat{\mathbf{C}}_{\mathbf{j}}(n) \right) \tag{7}$$

where $y(n)$ are the teacher data which we can construct from the original sequence; $\hat{(y)}(n)$ are the output of the ABE-WESN; $a_i, i = 1, 2, \cdots, e+1$ are the weight factors. We can calculate the weight factor by a optimal problem

$$\min_{a_1, \cdots, a_e, ae+1} \sum_{n=T_0+1}^{T} E(n) \tag{8}$$

For (8), we can obtain the solutions by (2.5)

$$A = \left(X^T X\right)^{-1} X^T Y \tag{9}$$

where $Y = [y(T_0), y(T_0+1), \cdots, y(T)]$, $A = [a_1, a_2, \cdots, a_e]$ and

$$\begin{bmatrix} \hat{\mathbf{y}}_{D_1}(T_0+1), & \hat{\mathbf{y}}_{D_2}(T_0+1), \cdots, & \hat{\mathbf{y}}_{D_e}(T_0+1), \hat{\mathbf{y}}_{C_e}(T_0+1) \\ \vdots & \vdots\vdots & \\ \hat{\mathbf{y}}_{D_2}(T), & \hat{\mathbf{y}}_{D_1}(T), \cdots, & \hat{\mathbf{y}}_{D_e}(T), \hat{\mathbf{y}}_{C_e}(T) \end{bmatrix}$$

Up to now, we have described the ABE-WESN model in detail. By (2.5) we can calculate the weight factor and further obtain the output of ABE-WESN.

3 Experiments

To verify the validation of the ABE-WESN, this section shows several experiments results. The experiments include both synthetic data and real-world engineering applications. First, we predict three challenging time series defined in [7], which are proposed on European Symposium on Time Series Prediction(ESTSP2008). Second, we use the proposed ABE-WESN models to predict the mobile traffic time series, which are from the real-life application and typically multiscale.

3.1 Experiments Setup

These section describes the setup of the experiments, including the performance measurements and the choice of the wavelet families.

Performance Measurements. The performance index used in this paper is the error of unbiased root mean square error(RMSE)

$$err = \sqrt{\frac{1}{(T - T_0 + 1)\sigma^2} \sum_{n=T_0+1}^{T} (\hat{\mathbf{y}}(n) - y(n))^2} \tag{10}$$

where $y(n)$ and $\hat{\mathbf{y}}(n)$ are the value of the original sequence and its corresponding estimation. σ is the variance of the sequence $y(n)$.

Choice of the Wavelet Families. One can choose different wavelet families, such as Haar, Daubechies, SymletsA, Biorthogonal, according to the properties of the original sequence. In this paper, we choose Dauberchies wavelet family according to the characteristics of the sequence.

3.2 Prediction of Time Series Defined on ESTSP 2008

Francis Wyffels et al presented an approach and solution to the European Symposium on Time Series Prediction 2008 challenge problem. The solution utilizes wavelet and ESNs architecture to model and predict all three time series defined in this conference. In this paper, we also deal with the challenging problem using the proposed ABE-WESN instead of WESN. For comparison, we carry out the experiments using the same setup for all three time series defined on ESTSP 2008. We repeat the experiments 100 times. Table 1 shows the average results. Fig. 2 illustrates the predicted result of the 3rd sequence.

Table 1. The predicting results of the ABE-WESN and WESN for the 3 time series defined on the ESTSP 2008. Trn Error and Tst Error denote training and testing ,respectively, which are calculated by (10).

Sequence	Model	Trn Samples	Tst Samples	Trn Error	Tst Error
I	ABE-WESN	331	18	0.591	0.4902
I	WESN	331	18	–	0.5000
II	ABE-WESN	1195	100	0.2501	0.3417
II	WESN	1195	100	–	0.3742
III	ABE-WESN	31409	200	0.1028	0.4804
III	WESN	31409	200	–	0.6481

We refer the predicted results of the WNESN model given in Table 1 to [7]. As we can see from Table 1, the proposed approach in this paper obtains better performance in terms of the performance index defined in (10). There are about $\%2, \%8.6, \%25$ percent enhancement, respectively.

3.3 Prediction of the Mobile Traffic Time Series

An accurate model and a prediction of mobile traffic play a crucial role in mobile network planning and design[10]. The data used in our experiments are from

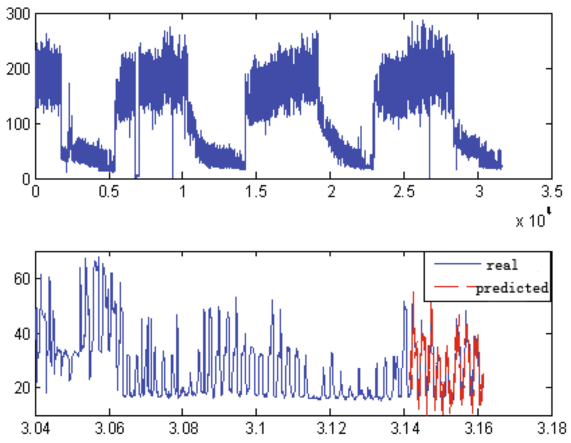


Fig. 2. Prediction Results of the 3rd time series using the proposed ABE-WESN. The top shows the original sequence. The bottom gives the comparison results of the practical and predicted results.

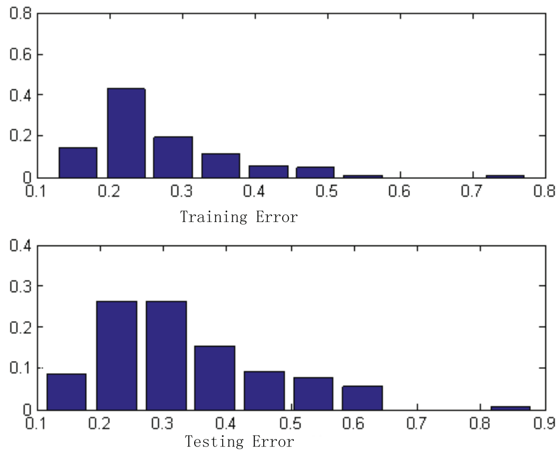


Fig. 3. The frequency of cells with NRMSE. we show the normalized root mean square error(NRMSE) on the horizontal axis and the frequency on the vertical axis. The top is of training error and the bottom is of the testing error.

the Network Monitoring System in China Mobile Communication Corporation Heilongjiang Co. Ltd. We extract the conversation traffic series from different 129 cells from July 1, 2008 to July 30, 2008. Note that we sample the conversation traffic every hour.

The conversation traffic sequences are typically multiscale time series and have several different periods. Thus, in the experiments, we choose 4 as the decomposing level and "db2" as wavelet family. The training and testing samples are

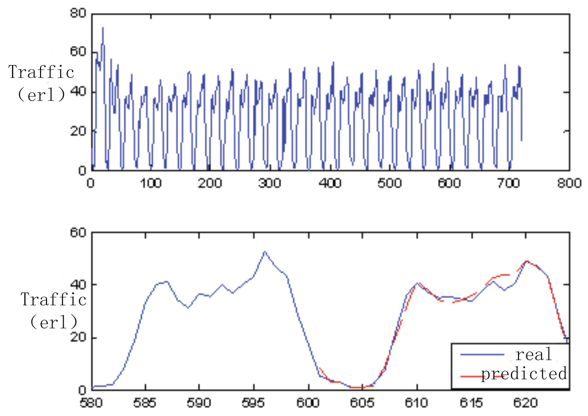


Fig. 4. The frequency of cells with NRMSE. The horizontal axis represents the normalized root mean square error(NRMSE). The vertical axis represents the frequency. The top is the training error and the bottom is the testing error.

480 and 24 ,respectively. The average training and testing error are 0.268(0.097) and 0.335(0.130)[1] .respectively. The histogram of training and testing errors of the 129 mobile cells are given in Figure 3. we can find that both training errors and testing error concentrate on the ranges of mean value. Note that we use the same parameters of the ABE-WESN in the previous experiments to different 129 mobile cells.This means that the proposed approach not only can model and predict the traffic time series accurately but also has strongly robust property. Figure 4 shows the training and testing results of a mobile cell.

4 Conclusion

To solve the multiscale problem, we proposed a novel approach, called ABE-WESN, based on the wavelet decomposition and echo state network. Considering the boundary effect and error cumulation of wavelet decomposition, we introduce weight factors. The weight factors make the final output of the model a linear combination instead of the sum of the different outputs of the ESN. Thus, this approach is totally different from previous approaches. The proposed approach not only models and predicts the multiscale time series accurately but also has a strong robust property. We carry out the experiments on both synthetic and real-world data. For synthetic data, we apply the proposed method to model and predict the 3 time series defined on the ESTSP 2008 and gain better performance. At the same time, we use the proposed method to solve a real-world engineering problem on how to predict the conversation traffic which is very key issue in network planning and design. The results show that the ABE-WESN can meet the requirements of modelling and predicting of the mobile conversation traffic.

[1] The data in the () are the variance.

References

1. Lukoševičius, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. Computer Science Review 3, 127–149 (2009)
2. Wang, S., Yang, X.-J., Wei, C.-J.: Harnessing nonlinear by sigmoid-wavelet hybrid echo state networks. In: The 6th World Congress on Intelligent Control and Automation, WCICA 2006, vol. 1, pp. 3041–3018 (2006)
3. Xue, Y., Yang, L., Haykin, S.: Decoupled echo state networks with lateral inhibition. Neural Networks 20(3), 365–376 (2007)
4. Wyffels, F., Schrauwen, B., Stroobandt, D.: Band-pass reservoir computing. In: Proceedings of the International Joint Conference on Neural Networks, pp. 3203–3208 (2008)
5. Holzmann, G., Hauser, H.: Echo state networks with filter neurons and a delay&sum readout. Neural Networks 23, 244–256 (2010)
6. Jaeger, H.: Discovering multiscale dynamical features with hierarchical Echo State Networks, Technical Report No. 10, Jacobs University Bremen (2007)
7. Wyffels, F., Schrauwen, B., Stroobandt, D.: Using reservoir computing in a decomposition approach for time series prediction. In: Proceedings of the European Symposium on Time Series Prediction, pp. 149–158 (2008)
8. Jaeger, H.: The "echo state" approach to analyzing and training recurrent neural networks. Techniqual Roprot GMD No. 148, German National Research Center for Information Technolgoy (2001)
9. Yu, P., Jianmin, W., Xiyuan, P.: Researches on Time Series Prediction with Echo State Networks. Acta Electronica Sinica 38(2A), 148–154 (2010)
10. Jianmin, W., Yu, P., Xiyuan, P.: Mobile Communication Traffic Forecast based on a New Fuzzy Model. In: Proceeding of International Instrumentation and Measurement Technology Conference (2009)
11. Shensa, M.J.: The Discrete Wavelet Transform: Wedding the À Trous and Mallat Algorithms. IEEE Transactions on Signals Processing 40(10), 2464–2482 (1992)

Research on Water Level Optimal Control of Boiler Drum Based on Dual Heuristic Dynamic Programming*

Qingbao Huang, Shaojian Song, Xiaofeng Lin, and Kui Peng

College of Electrical Engineering, Guangxi University,
530004 Nanning, China

Abstract. Boiler drum system is an important component of a thermal power plant or industrial production, and the water level is a critical parameter of boiler drum control system. Because of non-linear, strong coupling and large disturbance, it is difficult to reach a suitable working state of drum system by using traditional control methods. It is necessary to explore new methods to realize optimal control of drum water level. The back propagation (BP) neural network model of boiler drum system is built in this paper firstly, then the optimal control of the drum system by the dual heuristic dynamic programming (DHP) algorithm is realized, and compared with the heuristic dynamic programming (HDP) algorithm at last. The result shows that the DHP optimization algorithm has good performance in control precision and rejecting process disturbances.

Keywords: boiler drum level; BP neural network; dual heuristic dynamic programming; optimal control.

1 Introduction

Industrial boiler is an important equipment widely used in thermal power plant or industrial production. As the production equipment of high temperature and high pressure steam, which can drive turbine as power source and can be used as heat source for distillation and chemical reaction process as well, the drum system is the most important component of a boiler.

Water level is one of the main control parameters in boiler drum control system. On one hand, if the water level is too low, the water will evaporate quickly, then steam pressure will increase rapidly, which will affect the water circulation and threat to the safe of boiler system, even will cause a burst in steam pipe or result in catastrophic accident. On the other hand, if the water level is too high, it will lead to the steam bringing water into superheater and scaling in heating pipelines, which will cause superheater damaged. What's more, too high of water level will decrease the boiler efficiency and increase energy consumption [1]. So it is very important to keep drum water level within a desired range. However, because of the phenomenon of

* This work was supported by the Natural Science Foundation of China under Grant 60964002; the Natural Science Foundation of Guangxi Province of China under Grant 0991057; Scientific Research Foundation of GuangXi University under Grant XGL090017.

© Springer-Verlag Berlin Heidelberg 2011

False Water Level and the characteristic of typical nonlinear and strong coupling, it is difficult to obtain satisfying control quality with traditional control methods.

Many researchers have developed some control strategies such as two-impulse control and three-impulse control and other intelligent control for water level. For example, Fugang Huang [2] and Weijie Yue [3] *et al* researched adaptive drum water level based on fuzzy PID control, the stability was improved, but dynamic response was still slow. In 1977, Werbos proposed a new optimization method, that was adaptive dynamic programming (ADP). The method is very suitable to be used in hybrid, nonlinear and non-stationary environment. Chao Lu *et al* applied action-dependent heuristic dynamic programming (ADHDP) to a large power system stability control problem [4]. Derong Liu *et al* demonstrated a good engine torque and exhaust air-fuel ratio (AFR) control with adaptive critic techniques for an engine application [5]. Jianqiang Yi and Dongbin Zhao *et.al* formulated the ADP methods for ship course control problem and obtained fine tracking control result.

Because of good approximation capability to nonlinear system, BP neural network is used to build drum model based on offline data from a 75T/H industrial boiler in this paper. Then an optimal controller is designed with the dual heuristic dynamic programming algorithm to explore new method to stabilize the water level of boiler drum.

2 Adaptive Dynamic Programming (ADP)

2.1 Dynamic Programming [6]

Dynamic programming is based on Bellman's principle of optimality, which is a very useful tool in solving optimization and optimal control problems. Suppose that a nonlinear discrete-time system is given:

$$x(t+1) = f[x(t),u(t)], t = 0,1,2... \tag{1}$$

Where $x \in R^n$ represents the system state vector, $u \in R^m$ is the control action, f is the system function. Suppose the performance index of this system is

$$J[x(i),i] = \sum_{t=i}^{\infty} r^{t-i} U[x(t),u(t),t] \tag{2}$$

Where U is called the utility function, r ($0<r\leq 1$) is the discount factor. Note that the function J is dependent on the initial time i and the initial state $x(i)$, and it is referred to as the cost-to-go of state $x(i)$. The objective of dynamic programming problem is to choose a control sequence *u(t)*, *t* =*i*, *i*+1,…., so that the function J in (2) is minimized. According to Bellman equation, the optimal cost from time t is equal to

$$J^*[x(t),t] = \min_{u(t)}(U(x(t),u(t)) + \gamma J^*[x(t+1),t+1]) \tag{3}$$

The optimal control at time t is the *u(t)*, which achieves this minimum.

$$u^*(t) = \arg\min_{u(t)}(U(x(t),u(t)) + \gamma J^*[x(t+1)]) \tag{4}$$

2.2 Adaptive Dynamic Programming

In any case, it is often computationally untenable to run true dynamic programming due to the backward numerical process required for its solutions, which is the "curse of dimensionality". In 1977, Werbos [7] proposed an approach for ADP. To implement the ADP algorithm, Werbos proposed "approximate dynamic programming" formulation. The main idea of ADP shows in Fig. 1.

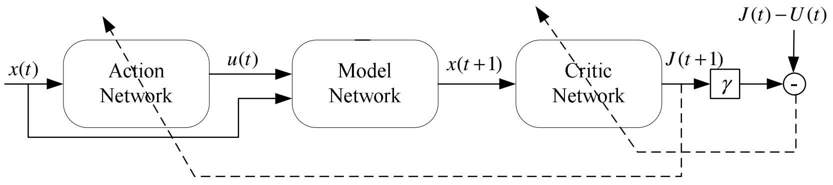


Fig. 1. The structure diagram of the ADP

The heuristic dynamic programming (HDP) and dual heuristic dynamic programming (DHP) are two main forms of adaptive dynamic programming. Of which, HDP has a critic network that estimates the function *J*(cost-to-go) in the Bellman equation of dynamic programming. DHP has a critic network that estimates the derivatives of *J* with respect to the vector $x(t)$. The critic network of DHP training is more complicated than in HDP since we need to take into account all relevant pathways of back propagation. Although the average training is much longer than HDP, DHP has better accuracy than the HDP. According to the characteristics of the boiler drum system, the optimal controller of boiler drum level is researched based on DHP algorithm in this paper [8].

3 Building the Neural Network Model of the Boiler Drum

3.1 The Analysis of Drum System

As the drum internal is a complex process, which has characteristic of typical nonlinear and strong coupling, it is difficult to find a suitable mathematical relationship to describe the model of the drum system. Since artificial neural networks can approximate to complex nonlinear system, we build the model of the drum system based on neural network.

According to the flow of steam measured by flow transmitter and water level measured by level transmitter last time, the water level of boiler drum can be changed by regulating water supply executor. As shown in Fig. 2, when the load is changed, the flow of steam is changed, then water supply are regulated before the water level fluctuations, in this way the water level will be stabilized within a reasonable scope [9].

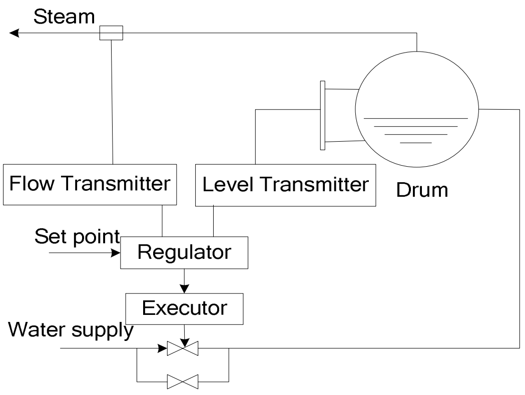


Fig. 2. Automatic drum level regulation system

3.2 Building the Model of Drum System

In this paper, model of the drum level system is built by BP neural network. For a drum level system, the control variable is: water supply, the state variables are: drum level and flow of steam. As shown in Fig.3, sm represents the input of the network model, $sm = [x(t)\ u(t)]$, where $x(t)$ is the state variables at time *t*, $u(t)$ is the control variable at time *t*, $x(t+1)$ is the actual output value of the model network, $\overline{x(t+1)}$ is the desired output value. In this paper, the structure of the model of the boiler drum system is 3-10-2. According to the structure of the boiler drum, the main controlled parameter is the drum water level, and the control parameter is water supply of the drum, while steam flow is determined by the load.

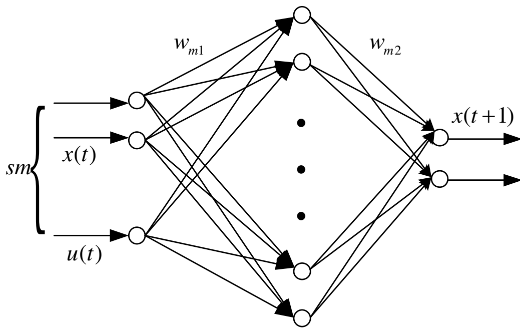


Fig. 3. The structure of BP neural network

Calculate model network forward, the transfer function is *tansig*:
Calculate input layer to hidden layer:

$$mh1 = w_{m1}{}^{T} * sm \tag{5}$$

Output of hidden layer is:

$$mh2 = \tan sig(mh1) \tag{6}$$

Calculate hidden layer to output layer:

$$v = w_{m2}{}^{T} * mh2 \tag{7}$$

Output of the model network is:

$$x(t+1) = \tan sig(v) \tag{8}$$

Define the error fuction of the model network:

$$e_m = d(t+1) - x(t+1) \qquad E_m = \frac{1}{2} e_m{}^2 \tag{9}$$

Where $d(t+1)$ is the desired output under the control signal $u(t)$, $x(t+1)$ is the actual output signal of the next network under the control signal $u(t)$ and state signal $x(t)$.

Using the gradient descent method adjust the weights of BP neural network.

$$\Delta w_{m2} = -l_m * \frac{\partial E_m}{\partial w_{m2}} = -l_m * \frac{\partial E_m}{\partial e_m} \frac{\partial e_m}{\partial w_{m2}} = -l_m * e_m * mh2 \tag{10}$$

$$\Delta w_{m1} = -l_m * \frac{\partial E_m}{\partial w_{m1}} = -l_m * e_m * \frac{\partial e_m}{\partial x(t+1)} * \frac{\partial x(t+1)}{\partial v} * \frac{\partial v}{\partial mh2} * \frac{\partial mh2}{\partial mh1} * \frac{\partial mh1}{\partial w_{m1}} \tag{11}$$

Where l_m =0.05 is the learning rate of model network.

Neural network model is trained based on 700 sets of offline historical data, which is collected from DCS of the boiler drum system. After data preprocessing, 400 sets of them are chosen as training samples, another 300 sets of them as the generalization ability testing samples of the model network. Part of them is shown in Table 1

Table 1. Part data of the drum system

NO.	flow of water supply （t/h）	flow of steam （t/h）	drum water level (mm)
1	80.586082	74.031746	40.952381
2	79.323563	74.622711	38.021976
3	78.598289	74.810745	29.084249
4	78.840051	74.864471	26.300365
…	…	…	…
700	64.442001	78.732597	14.285714

3.3 Model Testing and Results Analysis of Generalization Ability

The curve in Fig. 4 is the generalization ability curve of neural network model. The testing result shows that the model of the drum level system based on BP neural network has good generalization ability.

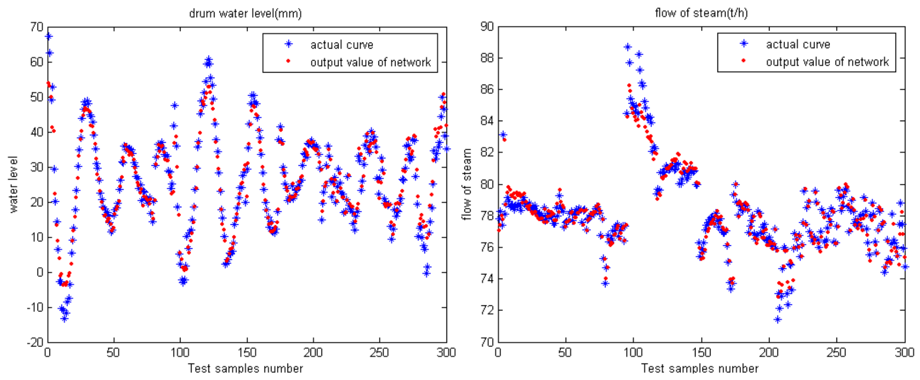


Fig. 4. Generalization ability of neural network model

4 Designing the Neural Network Controller Based on DHP Algorithm [10]

4.1 Training the Critic Network of DHP

According to the characteristic of input and output of the drum system, the controller of drum level system is designed by DHP optimization algorithm, which based on BP neural network. As shown in Fig. 1, the output of critic network is the derivatives of *J* with respect to the vector *x*(*t*), that is

$$\lambda(t) = \frac{\partial J(t)}{\partial x(t)} \tag{12}$$

The structure of critic network is: 1-10-1. The activation functions of hidden layer neuron and output layer neuron are *tansig* function and linear function respectively. The goal of critic network is to minimize the cost-to-go function $J_{\to\infty}$. According to the principle of dynamic programming, we can obtain

$$J(t) = U(t) + \gamma J(t+1) \tag{13}$$

$$\hat{\lambda}(t) = \frac{\partial J(t)}{\partial x(t)} = \frac{\partial U(t)}{\partial x(t)} + \frac{\partial U(t)}{\partial u(t)}\frac{\partial u(t)}{\partial x(t)} + \gamma \frac{\partial J(t+1)}{\partial x(t+1)}\frac{\partial x(t+1)}{\partial x(t)} \tag{14}$$

$$\lambda(t+1) = \frac{\partial J(t+1)}{\partial x(t+1)} \tag{15}$$

Where $U(t)$ is the utility function, γ=0.8 is discount factor.

The output of the critic network is $\lambda(t+1)$, the control strategy is evaluated by critic network. Reward or punishment is given by utility function, who gives a reword when the control result is more accurate. Define utility function by fixed-point tracking, $U(t) = (x(t) - \bar{x})$, where $\bar{x}$ represents control objective.
So we define the error function for the critic network

$$e_c(t) = \lambda(t) - \hat{\lambda}(t) \tag{16}$$

$$E_c = \frac{1}{2} e_c(t)^T * e_c(t) \tag{17}$$

The weights update rule for critic network is a gradient-based adaptation given by

$$\Delta w_{c2} = -l_c * \frac{\partial E_c}{\partial w_{c2}} = -l_c * \frac{\partial E_c}{\partial e_c} \frac{\partial e_c}{\partial w_{c2}} = -l_c * e_c * Jh2 \tag{18}$$

$$w_{c2} = w_{c2} + \Delta w_{c2} \tag{19}$$

$$\Delta w_{c1} = -l_c * \frac{\partial E_c}{\partial w_{c1}} = -\frac{\partial E_c}{\partial e_c} \frac{\partial e_c}{\partial w_{c1}} = -l_c * e_c * w_{c2}(1 - Jh2 * Jh2) * x(t+1) \tag{20}$$

$$w_{c1} = w_{c1} + \Delta w_{c1} \tag{21}$$

Where l_c=0.05 is the learning rate of critic network.

4.2 Training the Model Network of DHP

The model network is to approximate the system dynamic and it should be trained before the implementation of the DHP algorithm. The update rule of the model network is adopted as gradient decent method. The weights of the model network are kept unchanged after the network is trained.

4.3 Training the Action Network of DHP

The structure of action network is: 2-10-1. The activation functions of hidden layer neuron and output layer neuron are still *tansig* function and linear function respectively. The goal of action network is to find the optimal control signals $u^*(t)$, so as to minimize the cost-to-go function $J(t)$. According to optimal control equation, we can obtain

$$\frac{\partial J(t)}{\partial u(t)} = \frac{\partial U(t)}{\partial u(t)} + r\frac{\partial J(t+1)}{\partial u(t)} = \frac{\partial U(t)}{\partial u(t)} + r\frac{\partial J(t+1)}{\partial x(t+1)}\frac{\partial x(t+1)}{\partial u(t)} = 0 \tag{22}$$

So, we define the error function for the action network:

$$e_a(t) = \frac{\partial J(t)}{\partial u(t)} = \frac{\partial U(t)}{\partial u(t)} + r\frac{\partial J(t+1)}{\partial u(t)} = \frac{\partial U(t)}{\partial u(t)} + r\frac{\partial J(t+1)}{\partial x(t+1)}\frac{\partial x(t+1)}{\partial u(t)} \tag{23}$$

$$E_a = \frac{1}{2} e_a^2 \tag{24}$$

The weights update rule for action network is a gradient-based adaptation given by

$$\Delta w_{a2} = l_a * \frac{\partial E_a}{\partial w_{a2}} = l_a * e_a * \frac{\partial e_a}{\partial w_{a2}} \qquad w_{a2} = w_{a2} - \Delta w_{a2} \tag{25}$$

$$\Delta w_{a1} = l_a * \frac{\partial E_a}{\partial w_{a1}} = l_a * e_a * \frac{\partial e_a}{\partial w_{a1}} \qquad w_{a1} = w_{a1} - \Delta w_{a1} \tag{26}$$

Where l_a =0.03 is the learning rate of action network, r =0.9 is discount factor.

5 Simulation and Testing

For the critic network and the action network, the hidden layer uses the *tansig* function, and the output layer uses the linear function. The model network has been trained offline. The structure of the critic network and the action network are chosen as 1-10-1and 2-10-1 respectively. The control target of drum water level is 26mm and the initial weights of action network are constrained in (-1,1) randomly. We realized DHP algorithm and HDP algorithm and compared the result of them under the MATLAB environment. The result is shown in Fig. 5.

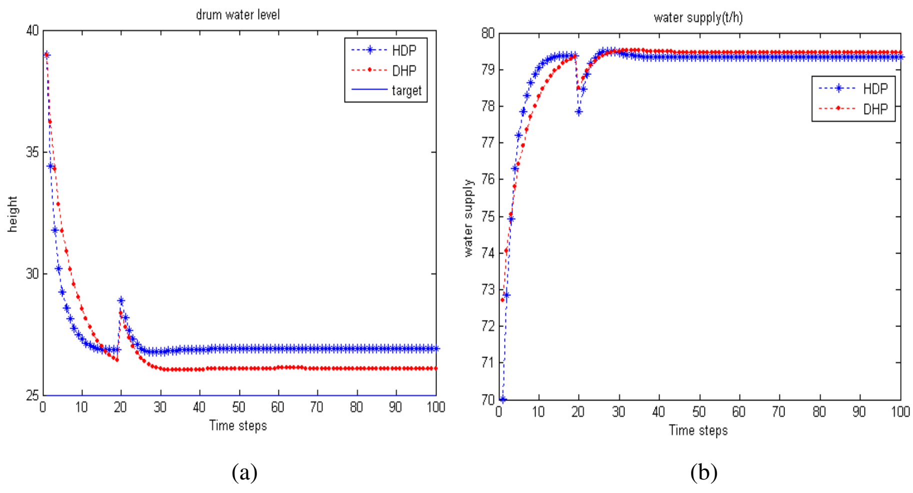


Fig. 5. Simulation results of DHP and HDP control algorithm

We display in Fig. 5(a) a typical trajectory of the drum level using the DHP algorithm. We conduct experiments starting from the initialized randomly inputs of action network. The experiment results show that, the controller which is based on DHP algorithm features its high speed of convergence and high control precision.

6 Conclusion

This paper is concerned with the drum level optimal control problem. Firstly, we analyze the boiler drum system and then build a BP neural network model based on the data from DCS. After that, the optimal control strategy of dual heuristic dynamic programming for water level system of boiler drum has been designed, whose structure and algorithm is made up of three BP networks. Through a large number of simulation studies have done, the optimal control strategy is more effective than the heuristic dynamic programming, which can adapt the changes of object parameters and shows a good quality on control with strong robustness, anti-jamming capacity and self-adaptability.

References

1. Sun, X., Tang, J., et al.: Practical computer control technology of boilers and industrial furnaces. National Defence Industry Press, Beijing (1993)
2. Huang, H., et al.: Based on Fuzzy-PID adaptive control of boiler drum water level. Electrical and Information (6) (2010)
3. Yue, W., Liu, Y.: Boiler drum level controlled by fuzzy self-adapting PID. In: Second Asia-Pacific Conference on Computational Intelligence and Industrial Applications, pp. 381–384 (2009)
4. Lu, C., Si, J., et al.: Direct heuristic dynamic programming for damping oscillations in a large power system. IEEE Trans. Syst., Man., Cybern. B 38(4), 1008–1013 (2008)
5. Liu, D., Javaherian, H., et al.: Adaptive critic learning techniques for engine torque and air–fuel ratio control. IEEE Trans. (2008)
6. Li, G.: Optimal control theory and parameter optimization. National Defence Industry Press, Beijing (2008)
7. Werbos, P.J.: Advanced forecasting methods for global crisis warning and models of intelligence. Gen. Syst. Yearbk. 22, 25–38 (1977)
8. Wang, F., Zhang, H., Liu, D.: Adaptive Dynamic Programming: An Introduction. IEEE Computational Intelligence Magazine (5) (2009)
9. Xin, G.: Boiler operation and operation guide. Machinery Industry Press (August 2006)
10. Wei, Q., Zhang, H., Liu, D.: An Optimal Control Scheme for a Class of Discrete-Time Nonlinear Systems with Time Delays Using Adaptive Dynamic Programming. Acta Automatica Sinica (2008)

Fuzzy Clustering-Based Polynomial Radial Basis Function Neural Networks (p-RBF NNs) Classifier Designed with Particle Swarm Optimization

Wook-Dong Kim, Sung-Kwun Oh, and Hyun-Ki Kim

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 polynomial-based Radial Basis Function Neural Networks (p-RBF NNs) classifier based on Fuzzy C-Means (FCM) clustering method. The parameters (fuzzification coefficient of FCM and polynomial type of models) are optimized by means of Particle Swarm Optimization (PSO). The fitness of hidden layer is expressed in term of partition matrix resulting from fuzzy clustering in this case being FCM. As weights between hidden layer and output layer, four types of polynomials are considered. The performance of proposed model is affected by some parameters such as the fuzzification coefficient of the fuzzy clustering (FCM) and the type of polynomial between hidden layer and output layer. The parameter coefficient of polynomial (weight) is obtained by using Weighted Least Square Estimation (WLSE) to improved performance and interprebility of local models. The proposed classifier is applied to a synthetic and machine learning dataset and its results are compared with those reported in the previous studies.

Keywords: Radial basis function neural network, Fuzzy C-means clustering, Weighted Least Square Estimation, Particle Swarm Optimization.

1 Introduction

In many pattern recognition systems, the paradigm of neural classifiers have been shown to demonstrate many tangible advantages with regard of criteria of learning abilities, generalization aspects, and robustness characteristic. Among these classifiers, multilayer perceptrons (MLPs) have been in a wide use. It is shown that the MLP can be trained to approximate complex discriminant functions. However, the MLP classifier requires a large number of parameters to be determined especially in case of a multilayer topology of this network. Also the number of iterations required to train these networks is often quite large. Radial Basis Function Neural Networks (RBF NNs) came as a sound alternative to the MLPs. RBF NNs exhibit some advantages including global optimal approximation and classification capabilities, and a rapid convergence of the learning procedures, see [1]. In spite of these advantages of RBF NNs, these networks are not free from limitations. In particular, discriminant functions

© Springer-Verlag Berlin Heidelberg 2011

generated by RBF NNs have a relatively simple geometry which is implied by the limited geometric variability of the underlying receptive fields (radial basis functions) located at the hidden layer of the RBF network. To overcome this architectural limitation, we introduce a concept of the polynomial-based Radial Basis Function Neural Networks (p-RBF NNs). Given the functional (polynomial) character of their weights connetcion in the P-RBF NNs, these networks can generate far more complex nonlinear discriminant functions.

In this paper, the fuzzy clustering-based p-RBF NNs classifier designed with the aid of FCM and the WLSE method involves structural and parametric optimization. As far as the structure optimization is concerned, there are three components to consider, i.e., the number of the cluster, fuzzification coefficient used in the FCM algorithm and the order of the polynomials that is equals to weights between hidden layer and output layer. These three components impact the performance of the proposed model and need to be optimized. In this paper, we carried out the structural as well parametric optimization by means of the Particle Swarm Optimization (PSO).

Section 2 describes the architecture of the FCM-based p-RBF NNs classifier and section 3 presents a learning method applied to the construction of proposed model. Section 4 deals with the PSO and the optimization of proposed model using the PSO. Section 5 presents the experimental results. Finally, some conclusions are drawn in Section 6.

2 Fuzzy Clustering-Based p-RBF Neural Networks

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 each node in hidden layer expresses a level of activation (fitness) of the receptive filed (radial basis function) given input variables. The node located at the output layer realizes a linear combination of the activation levels (fitness) and weights of single numeric values.

2.1 The Learning Method of Fuzzy C-Means in Hidden Layer

The proposed fuzzy clustering-based p-RBF neural networks comes as the extended structure of the conventional RBFNNs and the characteristic of the proposed model change hidden layer into FCM algorithm as shown in Fig. 1.

The hidden layer of proposed model used FCM algorithm divide input space as the number of cluster and the partition matrix of FCM is used as activation levels of receptive fields. Also, weights are not single numeric values but come in the form of polynomials involved input variables.

The weights of the fuzzy clustering-based p-RBF Neural Networks, four types of polynomials are considered. These are constant, linear, quadratic and modified quadratic as follows.

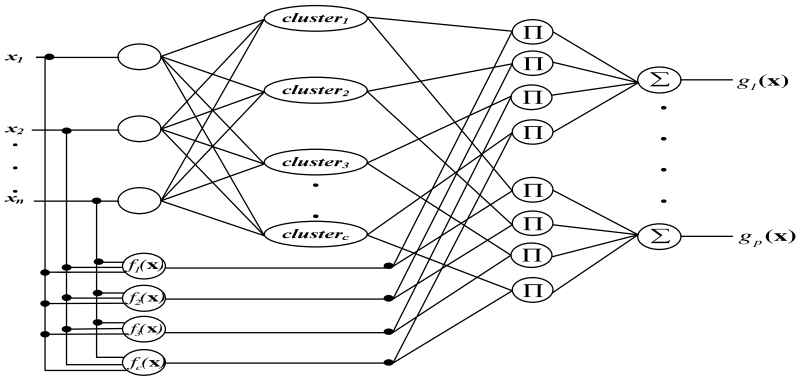


Fig. 1. Architecture of the fuzzy clustering-based p-RBF Neural Networks

Constant; $f_i(\mathbf{x}) = a_{i0}$ (1)

Linear; $f_i(\mathbf{x}) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j$ (2)

Quadratic; $f_i(\mathbf{x}) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j + \sum_{j=1}^{n} \sum_{k=j}^{n} a_{ijk} x_j x_k$ (3)

Modified Quadratic; $f_i(\mathbf{x}) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j + \sum_{k=1}^{n} a_{ijk} x_k^2$ (4)

These functions are activated by partition matrix and lead to local regression models located at the local model.

There are many different ways to describe pattern classifiers. One of the most useful ways is the one realized in terms of a set of discriminant functions $g_i(\mathbf{x})$, i=1,…,p (where p stands for the number of classes). The classifier is said to assign a input vector **x** to class ω_i
if

$$g_i(\mathbf{x}) > g_j(\mathbf{x}) \qquad \text{for all } j \neq i. \tag{5}$$

Thus, the classifiers are viewed as networks that compute p discriminant functions and select the category corresponding to the largest value of the discriminant.

In this paper, the proposed p-RBF NNs classifier is used as a discriminate function for two-class or multi-class. If a classification problem is multi-class one then we use (5) as discriminant function, otherwise, we use the following decision rule defined commonly as a single discriminant function $g(\mathbf{x})$ in two-class problem.

$$\text{Decide } \omega_1 \text{ if } g(\mathbf{x}) > 0\text{; otherwise decide } \omega_2. \tag{6}$$

2.2 The Learning Method of Fuzzy C-Means in Hidden Layer

We briefly review the objective function-based fuzzy clustering. The FCM algorithm is aimed at the formation of 'c' fuzzy sets (groups) in input space. The objective function Q is expressed as a sum of the distances of individual input data from the prototypes ($\mathbf{v}_1$, $\mathbf{v}_2$, …, $\mathbf{v}_c$).

$$Q = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{m} \|\mathbf{x}_k - \mathbf{v}_i\|^2 \tag{7}$$

Here, || || denotes a Euclidean distance method, '*m*' stands for a fuzzification coefficient, $m>1.0$. N is the number of patterns (data). The resulting partition matrix is denoted by $U=[u_{ik}]$, $i=1, 2,\dots,c$; $k=1, 2, \dots, N$. While there is a substantial diversity as far as distance functions are concerned, here we adhere to a weighted Euclidean distance taking on the following form

$$\|\mathbf{x}_k - \mathbf{v}_i\|^2 = \sum_{j=1}^{n} \frac{(x_{kj} - v_{ij})^2}{\sigma_j^2} \tag{8}$$

Where, σ_j denote a standard deviation of the j^{th} input variable. This type of distance is still quite flexible and commonly used.

Consider the set X which consists of N patterns treated as vectors located in some n-dimensional Euclidean space, that is, $X=\{\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_N\}$, $\mathbf{x}_k\in \mathbf{R}^n$, $1\le k\le N$. In clustering we assign patterns $\mathbf{x}_k\in X$ into c clusters, which are represented by its prototypes $\mathbf{v}_i\in \mathbf{R}^n$, $1\le i\le c$. The assignment to individual clusters is expressed in terms of the partition matrix $U = [u_{ik}]$ where

$$\sum_{i=1}^{c} u_{ik} = 1, \tag{9}$$

and

$$0 < \sum_{k=1}^{N} u_{ik} < N, \quad 1\le i\le c \tag{10}$$

The minimization of Q is realized in successive iterations by adjusting both the prototypes and entries of the partition matrix, that is min $Q(U, \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_c)$. The corresponding formulas used in an iterative fashion read as follows

$$u_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{\|\mathbf{x}_k - \mathbf{v}_i\|}{\|\mathbf{x}_k - \mathbf{v}_j\|}\right)^{\frac{2}{m-1}}} \tag{11}$$

and

$$\mathbf{v}_i = \frac{\sum_{k=1}^{N} u_{ik}^{m}\mathbf{x}_k}{\sum_{k=1}^{N} u_{ik}^{m}}, \tag{12}$$

We note that the resulting partition matrix produces '*c*' fuzzy sets with the elements of the matrix $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_c$ forming the corresponding rows of the partition matrix U, that is $U = [\mathbf{u}_1^T\ \mathbf{u}_2^T \dots \mathbf{u}_c^T]$. From the design standpoint, there are several essential parameters of the FCM that impacts its usage of the produced results. These parameters concern the number of clusters, the values of the fuzzification coefficient and a form of the distance function. The fuzzification coefficient exhibits a significant impact on the form (shape) of the developed clusters. The commonly used value of "*m*" is equal to 2. Lower values of the fuzzification coefficient produce more Boolean-like shapes of the fuzzy sets where the regions of intermediate membership values are very much reduced. When we increase the values of "*m*" above 2, the resulting membership functions start to become "spiky" with the values close to 1 in a very close vicinity of the prototypes.

2.3 The Learning Method of WLSE

The weights between hidden layer and output layer are concerned with the estimation of the parameters of the polynomial of the local model. The main difference between the WLSE and standard LSE is the weighting scheme which comes as a part of the WLSE and makes its focus on the corresponding local model. The learning method of standard LSE is oriented to minimize its overall squared error between real output and model output, therefore 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.

$$Error(E) = \sum_{i=1}^{n}\sum_{k=1}^{m} u_{ik}(y_k - f_i(\mathbf{x}_k - \mathbf{v}_i))^2 \tag{13}$$

Where, u_{ik} is the partition matrix $\mathbf{u}_1$ (activation level) of the i^{th} cluster

The performance index J_L can be re-written using matrix notation

$$\begin{aligned} E &= \sum_{i=1}^{n}(\mathbf{Y} - \mathbf{X}_i\mathbf{a}_i)^T \mathbf{U}_i(\mathbf{Y} - \mathbf{X}_i\mathbf{a}_i) \\ &= \sum_{i=1}^{n}\left(\mathbf{U}_i^{1/2}\mathbf{Y} - \mathbf{U}_i^{1/2}\mathbf{X}_i\mathbf{a}_i\right)^T\left(\mathbf{U}_i^{1/2}\mathbf{Y} - \mathbf{U}_i^{1/2}\mathbf{X}_i\mathbf{a}_i\right) \end{aligned} \tag{14}$$

Where, $\mathbf{a}_i$ is the vector of coefficient of polynomial (local model), is $\mathbf{Y}$ the vector of output data, $\mathbf{U}_i$ 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 polynomial function is linear, $\mathbf{X}_i$ and $\mathbf{a}_i$ can be expressed as follows

$$\mathrm{U}_i = \begin{bmatrix} u_{i1} & 0 & \cdots & 0 \\ 0 & u_{i2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{in} \end{bmatrix} \qquad \mathbf{X}_i = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1l} \\ 1 & x_{12} & \cdots & x_{1l} \\ 1 & \vdots & \ddots & \vdots \\ 1 & x_{1n} & \cdots & x_{ln} \end{bmatrix} \qquad \mathbf{a}_i = [a_{i0}\ a_{i1} \cdots a_{il}]$$

For the local learning algorithm, the objective function is defined as a linear combination of squared error, which is a difference between real (desired) output and the output of local model when considering the weighting factor matrix $\mathbf{W}_i$. This matrix captures the activation levels with respect to the ith fuzzy cluster. In this sense, we can consider the weighting factor matrix as activation levels of the corresponding cluster. The optimal coefficients of the polynomial of ith cluster are described as follows

$$\mathbf{a}_i = (\mathbf{X}_i^T \mathbf{U}_i \mathbf{X}_i)^{-1} \mathbf{X}_i \mathbf{U}_i \mathbf{Y} \tag{15}$$

Notice that the coefficients of the polynomial of each local model 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 cluster.

3 Optimization Process of Fuzzy Clustering-Based p-RBF NNs

The underlying principle of the PSO involves a population-based search in which individuals representing possible solutions carry out a collective search by exchanging their individual findings while taking into consideration their own experience and evaluating their own performance. PSO involves two competing search strategy aspects. First, the individuals ignore their own experience and adjust their behavior according to the successful beliefs of individuals occurring in their neighborhood. Second, the *cognition* aspect of the search underlines the importance of the individual experience where the individual is focused on its own history of performance and makes adjustments accordingly. PSO is conceptually simple, easy to implement, and computationally efficient. Unlike many other heuristic techniques, PSO has a flexible and well-balanced mechanism to enhance the global and local exploration abilities. The particle swarm optimization (PSO) is applied to parametric optimization such as the number of cluster, the fuzzification coefficient and Polynomial type.

4 Experimental Results

To evaluate the performance of our model, we used a synthetic datasets and Machine Learning datasets. Our objective is to quantify the performance of the proposed model and compare it with the performance of some other classifiers reported in the literature. In the assessment of the performance of the classifiers, we report the % of correctly classified patterns (classification rate).

4.1 Synthetic Datasets

We start with a series of two dimensional synthetic datasets as shown in Figure 2.

This dataset is divided into three classes having each class consists of 100 pattern. We carried out the experiment about synthetic datasets, Case I is non-optimization model that just changed the number of the cluster from 2 to 5 and Case II is optimized model using PSO.

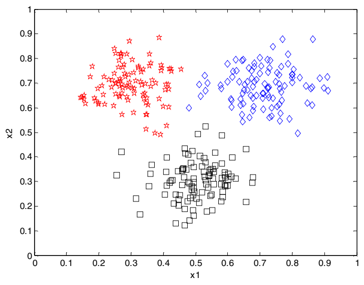


Fig. 2. Synthetic datasets (two inputs – three classes)

In Table 1, the number for each model indicates the success rate for recognition and the number in parenthesis describe the total number of classification error.

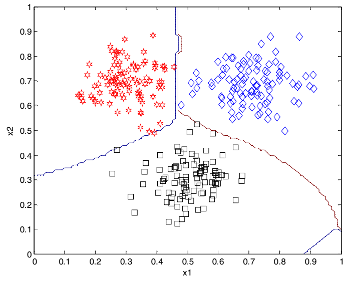
Table 1. Classification Rate for synthetic datasets

Model	No. of clusters	Fuzzification Coefficient	Polynomial type	Classification Rate [%]
Case I *Without optimization*	2	2	Linear	99.00 (3)
			Quadratic	99.33 (2)
			M. Quadratic	99.67 (1)
	3	2	Linear	99.33 (2)
			Quadratic	99.33 (2)
			M. Quadratic	99.33 (2)
	4	2	Linear	99.33 (2)
			Quadratic	99.33 (2)
			M. Quadratic	99.33 (2)
	5	2	Linear	99.00 (3)
			Quadratic	99.33 (2)
			M. Quadratic	99.33 (2)
Case II *With optimization*	4	1.19	M. Quadratic	100.00 (0)
	5	1.10	Quadratic	100.00 (0)

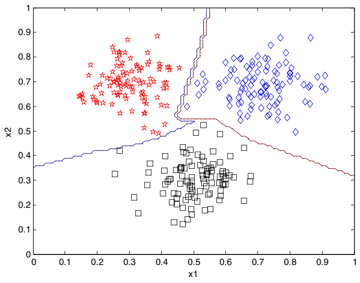
Figure 3 (a) and (b) shows the classification boundary and partitions in case of Case II, and (c) and (d) denote boundary and partitions of Case I corresponding Case II. Figure 3 visualize the classification boundaries when Classification Rate is 100.00%, 100.00%, 99.33%, and 99.33% respectively, In the all case, the performance of proposed model with optimization (Case II) is better than its without optimization (Case I).

4.2 Machine Learning Dataset

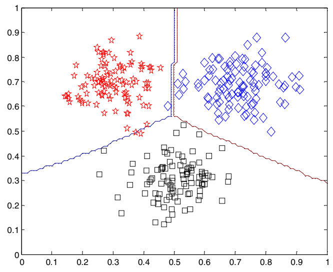
We consider glass datasets concerning classification problems coming from the Machine Learning repository (http://www.ics.uci.edu/~mlearn/MLRepository.html). This



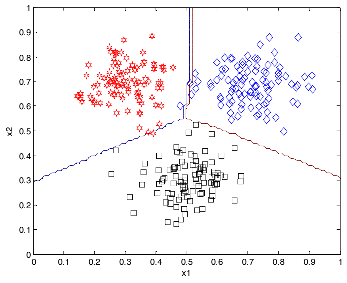
(a) Case II (No. of cluster 4)



(b) Case II (No. of cluster 5)



(a) Case I (No. of cluster 4)

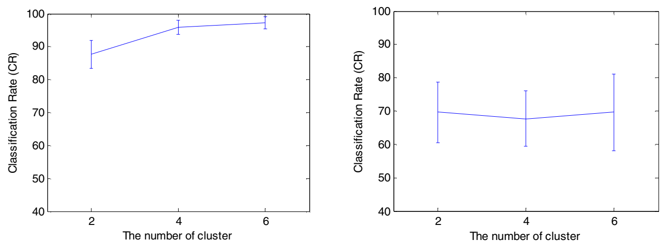


(b) Case I (No. of cluster 5)

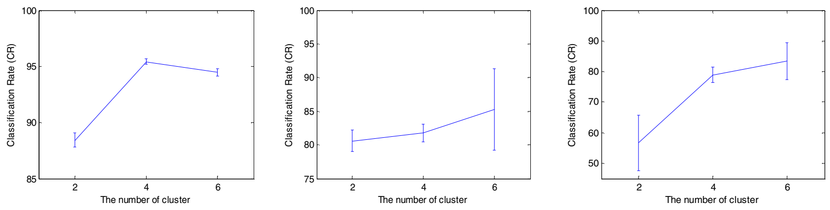
Fig. 3. Classification Boundaries of Classifiers

Table 2. Classification Rate for glass dataset

No. of cluster		Fuzzification Coefficient	Polynomial type	Training CR	Validation CR	Testing CR
2	2 split	1.10 ~ 4.15	All Quadratic	87.74 ± 04.18		69.65 ± 9.08
	3 split	1.10 ~ 2.15	All Quadratic	88.42 ± 0.64	80.59 ± 1.61	56.67 ± 9.13
4	2 split	1.10 ~ 1.51	Quadratic or M. Quadratic	95.91 ± 2.23		67.74 ± 8.37
	3 split	1.17 ~ 1.18	Quadratic or M. Quadratic	95.44 ± 0.26	81.76 ± 1.32	78.89 ± 2.48
6	2 split	1.10 ~ 1.48	All Quadratic	97.20 ± 1.91		69.63 ± 11.60
	3 split	1.19 ~ 1.20	Quadratic or M. Quadratic	94.50 ± 0.32	85.29 ± 6.09	83.33 ± 6.09



(a) 2split datasets (Training, and Testing)



(b) 3split datasets (Training, Validation, and Testing)

Fig. 4. Classification rate and standard deviation versus the number of cluster

dataset includes 214 input-output pairs. The number of input variables is 9 (after removing index number) and outputs consist of 6 class. In this experiment, we divided into type I) 2plit datasets (training and testing) and type II) 3split datasets (training, validation, and testing). The five-fold cross-validation (CV) is used and the final classification rate is the average of 5-fCV.

Table 2 describes the Classification Rate (CR) for recognition according to the number of cluster. Table 2 shows classification rate of training, validation, and testing versus the number of cluster. The performance of testing in case of 3split is better than in case of 2 split. Figure 4 describes average classification rate and standard deviation using error-bar in MATLAB.

Table 3. Comparison of classification rate reported in in the experiments

Models		**Classification Rate of Testing dataset [%]**
Reported in the literature		
Tabu Search/*k*-Nearest Neighbor [5]		81.4
Locally Adaptive Metric Nearest Neighbor [6]		75.2
Decision Tree Method [6]		68.2
Bayesian networks [6]		74.4
Proposed model		
3split	Cluster : 4	78.89
	Cluster : 6	83.33

5 Concluding Remarks

In this paper, we have proposed fuzzy clustering-based polynomial radial basis function neural network classifier. The proposed model is the extended architecture of conventional RBFNN. The classification rate of fuzzy clustering-based p-RBF NNs classifier is affected by the type of polynomials as well as some parameters such as the number of cluster and fuzzification coefficient of FCM algorithm. The PSO is exploited to find the structural as well as parametric factors minimizing classification rate of the proposed model.

Acknowledgements. This work(Grants No. 00043401) was supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded by Korea Small and Medium Business Administration in 2010 and also 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 SUWON2010-B2, Center for U-city Security & Surveillance Technology].

References

1. Park, P.J., Oh, S.K., Kim, H.K.: Design of Polynomial Neural Network Classifier for Pattern Classification with Two Class. Fuzzy Set and Systems 145, 165–181 (2008)
2. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proc. IEEE Int. Confrence Neural Networks, vol. 4, pp. 1942–1948 (1995)
3. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
4. Duda, R.O., Hart, P.E.: pattern Classification and Scence Analysis. Wiley, New York (2002)
5. Pernkopf, F.: Bayesian Network Classifiers versus Selective k-NN classifier. Pattern Recognition 38, 1–10 (2005)
6. Tahir, M.A., Bouridane, A., Kurugollu, F.: Simultaneous Feature Selection and Feature Weighting suing Hybrid Tabu Search/k-nearest Neighbor Classifier. Pattern Recognition Letters 28, 438–446 (2007)

Application of RBF Neural Network and Nonlinear Particle Filter in the Synthetic Ammonia Decarbornization

Yongwei Li, Jia Zhong, Tao Yuan, and Ying Zhang

College of Electrical Engineering and Information Science,
Hebei University of Science and Technology,
050018 Shijiazhuang, Hebei, China
{0599li,zhongjiabbm}@163.com, 675573698@qq.com,
yingskylove@126.com

Abstract. The synthetic ammonia decarbornization industrial process is a complex production process with strong nonlinearity, large delay and strong coupling. It is difficult to set up the on-line control model of the process. The drawback of the conventional BP neural network algorithm used to building system modeling is easily falling into the minimum value. This paper is concerned with the use of a RBF (Radial Basis Function) neural network control based on particle filter algorithm to solve above problems. The RBF neural network can approximate any continuous function and the particle filter can deal with nonlinear problems. This approach could deal with a complex multi-phase system. The method introduced in the paper is to set up a RBF neural network control model firstly, and then, the weights of RBF neural network are optimized by the particle filter algorithm. Compared to the fuzzy neural network which is applied, the simulation result of the method in this paper demonstrates that the control accuracy and system response speed are improved significantly.

Keywords: RBF neural network; nonlinear particle filter; the synthetic ammonia decarbornization.

1 Introduction

The synthetic ammonia decarburization industrial process is a complicated process with nonlinearity, large delay and strong coupling. The complex system has a lot of uncertainties, e.g. lag, randomness, fuzzy, gray. So an accurate model is difficult to be established [1]. Some traditional methods are used in the complex system, such as fuzzy control, fuzzy-neural network control, neural network forecast. They play a definite effect. However, as for the complex systems, these traditional methods are only applied to some particular systems. It is a hot point problem and a Gordian knot in the field of process control that how to control the complex system effectively. The RBF neural network has the advantages, e.g. fast convergence, simple construct and approximating any continuous function. Using RBF neural network for modeling and

© Springer-Verlag Berlin Heidelberg 2011

control is an available way. A nonlinear and non-Gauss system can be optimized by particle filter algorithm. This algorithm is not restricted by Gauss characteristics and is applied to the nonlinear and non-Gauss system. The weights of RBF neural network is optimized by the particle filter algorithm. Building RBF neural network structure model based on non-linear particle filter is a method of control the complicated process effectively. An effective way is provided to solve a class of complex systems modeling and optimization control.

2 Nonlinear Particle Filter

The method of dealing with state estimation of the nonlinear non-Gaussian system includes Unscented Kalman Filter (UKF) and Extended Kalman Filter (EKF). But they are restricted by the linear Kalman filter algorithm which should meet the Gaussian distribution. The accuracy of the nonlinear system state estimation is lower by linearization. States of complex systems are estimated by nonlinear particle filter algorithm. The accuracy can be improved and the more system state variables prior information can be obtained. The nonlinear particle filter algorithm is an appropriate method for non-Gaussian state estimation.

Particle filter algorithm can achieve recursive Bayesian filtering by non-parametric Monte Carlo method. It can be applied in state estimation of nonlinear systems[2]. Its principle is that finding a group of random sample spreading in the state space, approximating probability density function by sample means instead of integral operator, obtaining the minimum variance estimation of states. Nonlinear particle filter has a unique advantage of parameter estimation and station filtering of nonlinear non-Gaussian time-varying systems [3], [4], [5]. At present, the nonlinear particle filter has been applied to target tracking [6], signal processing [7] and other fields. However, application of it in the process control is not reported.

We supposed that the dynamic time-varying systems are described as follows:

$$\mathrm{x}_k = f_{k-1}\left(x_{k-1}, v_{k-1}\right) . \tag{1}$$

$$z_k = h_k\left(x_k, n_k\right) . \tag{2}$$

By transition probability matrix form, the system can be described as:

$$P\left(\mathrm{x}_k \middle| x_{k-1}\right) = \oint\left(\mathrm{x}_k - f_{k-1}\left(x_{k-1}, v_{k-1}\right)\right)P\left(v_{k-1}\right)dv_{k-1} . \tag{3}$$

$$P\left(\mathrm{z}_k \middle| x_k\right) = \oint\left(\mathrm{z}_k - h_k\left(x_k, n_k\right)\right)P\left(n_k\right)dn_k . \tag{4}$$

Chapman-kolmogorov equation

$$P\left(\mathrm{x}_k \middle| \mathrm{z}_{1:(k-1)}\right) = \oint\left(\mathrm{x}_k \middle| \mathrm{x}_{k-1}\right)P\left(\mathrm{x}_{\mathrm{k-1}} \middle| z_{1:(\mathrm{k-1})}\right)dx_{k-1} . \tag{5}$$

And the probability density function $P\left(x_{k\text{-}1} \middle| z_{1:(k\text{-}1)}\right)$ uses the re-sampling method to gain number T random sample points $\left\{x_{k-1}^{\mathrm{i}}\right\}_{i=1}^{\mathrm{T}}$.

If the sample sizes are enough, then

$$P\left(\mathrm{x}_{\mathrm{k-1}}\middle| z_{1:(\mathrm{k-1})}\right)=\sum_{i=1}^{T}\omega_{k-1}^{i}\delta\left(x_{k-1}-x_{k-1}^{i}\right) . \tag{6}$$

The prediction equation is

$$P\left(\mathrm{x}_{\mathrm{k-1}}\middle| z_{1:(\mathrm{k-1})}\right)=\sum_{i=1}^{T}\omega_{k-1}^{i}P\left(x_{k}\middle| x_{k-1}^{i}\right) . \tag{7}$$

The state update equation is

$$P\left(\mathrm{x}_{k}\middle| z_{1:k}\right)\approx\sum_{i=1}^{T}\omega_{k}^{i}\delta\left(x_{k}-x_{k}^{i}\right) . \tag{8}$$

where

$$\omega_{k}^{i}=\omega_{k-1}^{i}\frac{P\left(z_{k}\middle| x_{k}^{i}\right)P\left(x_{k}^{i}\middle| x_{k-1}^{i}\right)}{q\left(x_{k}^{i}\middle| x_{k-1}^{i},z_{k}\right)} . \tag{9}$$

The key of nonlinear particle filter is the selection of the important probability density function and re-sampling.

1) The important probability density function is selected by sub-optimal algorithm.

2) Re-sampling is a method which can solve the deficiency of particle number. The basic idea is to use particles and the weight probability density function to increase re-sampling steps between two samples. In other words, it means increasing the number of particles which have larger weight and reducing the number of particles which have small weight. Theoretically, such a particle size distribution will approximate the true state posterior probability distribution. In practice, random sampling method is regard as re-sampling. The schematic diagram of re-sampling is shown in figure 1.

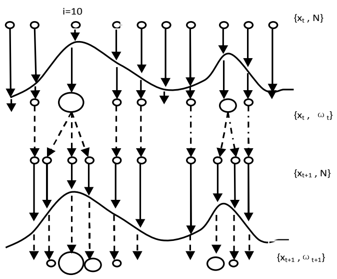


Fig. 1. Schematic diagram of the re-sampling

The process of random sampling is: first, generate random numbers from uniform distribution[0, 1]; second, find a set of integer values, which makes $\sum_{j=0}^{m-1} h_j < \mu_l \le \sum_{j=0}^{m} h_j$ and the sample $x_k^{(m)}$ as a new sampling; according to $\lambda_j = \sum_{j=0}^{i} h_j, i = 1,...,n$, break up the interval [0, 1] into n small intervals; when random number μ_l is in the m interval, $I_m = (\lambda_{m-1}, \lambda_m]$, copy the sample $x_k^{(m)}$.

From the random sampling process, the samples which have large weights are copied several times and the samples which have very small weight are given up. The total particle number keeps a constant. The prior probability in time t is expressed by the particles with weight. Through systematic observation and re-calculating the weight, the particle which has large weight is split into new particles. Other particle which has small weight is given up. Therefore, a new set of particles are obtained. This method makes the particle size distribution approximate the true state posterior probability distribution. Meanwhile, the computational complexity and the computational workload are reduced. When the particles are mixed with random, the state in the next time (t+1) can be forecasted. It is a system state transition process. When we enter systematic observation process, we can forecast the state in the next time [8], [9].

Nonlinear particle filter algorithm implementation steps are: firstly, number T of finite samples is randomly selected from $q(x_k \mid x_{k-1}, z_k)$; secondly, the corresponding $P(x_k \mid x_{k-1})$ and $P(z_k \mid x_k)$ are calculated point by point; thirdly, importance weight coefficient is calculated sample by sample, using formula (9); fourthly, weights are normalized by formula (10); fifthly, $P(x_k \mid z_{1:k})$ is estimated by formula (8).

$$\tilde{\omega}_k^i = \frac{\omega_k^i}{\sum_{j=1}^{n} \omega_k^j} . \tag{10}$$

3 Improved RBF Neural Network

3.1 RBF Neural Network Model

Artificial neural network has ability on nonlinear mapping. It can generalize automatically the function relation between the data through learning or training. Therefore, it is suitable for effective modeling. Currently, the BP neural network based on the gradient descent algorithm is used widely. However, when the BP neural network approximates a continuous function, it always falls into local minima and converges slowly. The RBF neural network has the characteristics of simple construct and training concise. Moreover, it can approximate any continuous function. It is better in approximation capability, sorting capability and learning speed than BP neural network. RBF neural network is widely used in time series analysis, nonlinear control, pattern recognition and image processing. Therefore, RBF neural network can be used in system modeling and process control [10], [11], [12].

RBF neural network is a three-layer forward neural network, which consists of three layers: input layer, hidden layer and output layer. The RBF neural network structure is 3 inputs and 1 output. The input layer nodes (the number of nodes is i) are formed by input signals. Hidden layer (the number of nodes is j) is the middle layer. Its excitation function is a non-negative non-linear basis function, which is symmetrical and damping about center. Basis function generates response in some part. For the theoretical analysis, basis function is Gaussian function usually:

$$u_j = exp\left[-\frac{\left(X - C_j\right)^{\mathrm{T}}\left(X - C_j\right)}{2\sigma_j^2} \right], j = 1, 2, \cdots, m \ . \tag{11}$$

where u_j is the output of hidden layer node j; $X = \left(x_1, x_2, \cdots, x_n\right)^{\mathrm{T}}$ are n-dimensional input samples; C_j is center vector of hidden layer nodes j; m is the number of hidden nodes. The third layer is output layer that usually has a linear function. It is a linear combination in the output of hidden layer nodes.

RBF neural network has basic working principles: hidden layer is a product of weighted inputs and the corresponding thresholds. In addition, the network output of hidden layer is obtained by Radial Basis Function. Weighted input is the distance between input layer input vector and the weight vector. It is obtained by distance function. The input of output layer is the sum of weighted input and the corresponding threshold. The network output is obtained by linear function. Weighted input is a product of the hidden layer output vector and weight vector.

If a set of training samples is given, the key of creating a RBF neural network is confirmation of center vector C_j, variance σ_j and weight ω_j by samples. The radial basis network is trained by two steps. Firstly, use the unsupervised K-means clustering algorithm to discriminate RBF parameters of hidden layer excitation function. The number of hidden nodes, RBF neural network center vector C_j and variance σ_j is ascertained. Then, the output layer linear weight ω_j is calculated by error correction learning algorithm.

3.2 Improvement of RBF Neural Network

The radial basis function center is usually ascertained by the unsupervised K-means clustering algorithm. This method is simple and has fast convergence. But this algorithm is too sensitive to initial conditions. If the parameters of initial nodes are set incorrectly, some nodes, which are not trained, are dead nodes. So this paper uses improved HCM algorithm (IHCM algorithm) to ascertain the radial basis function center. Expanded vector is obtained by combination of input vector and output vector to ascertain the radial basis function center. Assume the following expanded vector:

$$\begin{cases} r_q^{\mathrm{I/O}} = \left[X_q^T \; Y_q^T \right]^T \\ \left\{X_q\right\}_{q=1}^{K} \in \mathrm{R}^N \\ \left\{Y_q\right\}_{q=1}^{K} \in \mathrm{R}^M \end{cases} . \tag{13}$$

The sample set $R = \{r_1, r_2, \cdots, r_k\}$ is divided into N_R classes. Any expanded vector in set R belongs to a class entirely. And any class has a sample. The classification result is expressed by matrix $U(N_R * K)$.

The element of U is

$$u_{iq} = \begin{cases} 1 & \text{when } r^{\mathrm{I/O}} \in \mathrm{A}_i \\ 2 & \text{when } r^{\mathrm{I/O}} \notin \mathrm{A}_i \end{cases} . \tag{14}$$

where $\mathrm{A_i} =(1,2,\cdots,N_R)$ is sample set i.

The concrete steps are:

1) The number N_R of cluster category is obtained. $2 \le N_R \le \mathrm{K}$, K is the number of samples.

2) Maximum permissible error ($\mathrm{E_{max}}$) is set.

3) Initial classification matrix U^t is appointed. t=0, t is iterations.

4) Expanded center vector g_j is calculated by matrix U^t and formula (12).

$$g_j = \frac{1}{\sum_{\mathrm{q}=1}^{\mathrm{K}} u_{\mathrm{iq}}} \sum_{\mathrm{q}=1}^{\mathrm{K}} u_{\mathrm{iq}} r_{\mathrm{q}}^{\mathrm{I/O}} . \tag{15}$$

5) Matrix U^t was updated to U^{t+1} by the following rule.

$$u_{iq}^{t+1} = \begin{cases} 1 & \text{when } d_{iq}^t = \min\limits_{1 \le j \le N_R} \{d_{jq}^t\} \\ 0 & \text{others} \end{cases} . \tag{16}$$

where $d_{\mathrm{jq}} = \left\| r_{\mathrm{q}}^{\mathrm{I/O}} - g_{\mathrm{j}} \right\|$ stands for the Euclidean distance between expanded vector $r_{\mathrm{q}}^{\mathrm{I/O}}$ and expanded center vector g_{j}.

6) Matrix U^t and U^{t+1} are compared by a matrix norm ε. If $e > \left\| \mathrm{U}^t - \mathrm{U}^{t+1} \right\|$, it stops. Otherwise, t=t+1, back to step (4).

In IHCM algorithm, expanded center vector g_{j} is obtained, $g_j \in R^{\mathrm{N+M}} (j = 1, 2, \cdots, N_{\mathrm{R}})$. The center of RBF neural network vector C_j is the ahead N-dimensional expanded center vector g_{j}, namely $\mathrm{C}_j = \mathrm{g}_j^{\mathrm{x}} \in \mathrm{R}^{\mathrm{N}}$.

4 Weights of RBF Neural Network Is Optimized by the Nonlinear Particle Filter Algorithm

The weights of RBF neural network are optimized by the nonlinear particle filter algorithm. First, the RBF neural network model is created, and then the weights of RBF neural network are optimized by the nonlinear particle filter algorithm. The optimization weights are obtained by re-sampling [13], [14].

RBF neural network system equations are:

$$\mathrm{w}_k = \mathrm{w}_{k-1} + \mathrm{v}_{k-1} \ . \tag{17}$$

$$y_k = \sum_{j=1}^{n} w_j \mathrm{sig} \sum_{i=1}^{m} x_i w_{ji} + u_k \ . \tag{18}$$

where $w_{ji}\left(i=1,2,\ldots,m;\ j=1,2,\ldots,n\right)$ is the weight between input layer and hidden layer; $w_j\,(j=1,2,\ldots,n)$ is the weight between hidden layer and output layer; x_i is system input; y_k is system output (the measurement of system); Observation function is $G(W_k, x_k) = \sum_{j=1}^{n} w_j \mathrm{sig} \sum_{i=1}^{m} x_i w_{ji}$; V_{k-1} is system noise; u_k is observation noise, noise is normal distribution; *k* is time. Put sample data into the particle filter algorithm and optimize the weights of RBF neural network. After several iterations, the optimization weights are obtained. The flow chart is shown in Fig. 2.

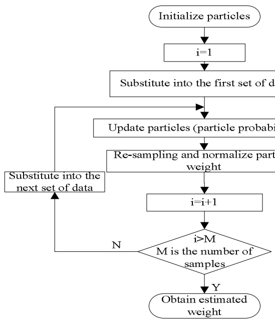

Initialize particles
i=1
Substitute into the first set of data
Update particles (particle probability)
Re-sampling and normalize particle weight
Substitute into the next set of data
i=i+1
i>M
M is the number of samples
N
Y
Obtain estimated weight

Fig. 2. Flow chart of RBF neural network weights optimization based on particle filter

5 Application in the Synthetic Ammonia Decarburization

The core technology of the synthetic ammonia decarburization conducted in carbonation towers. The reaction of carbonation tower is the three-phase system with gas, liquid, solid. It is also a system with complex absorption, reaction, crystallization and heat transfer. Therefore, in the coalition of carbonization process, there is mutual coupling in the flow of materials with serious internal and external interference. From the control point, carbonation process is a multi-phases, multi-variables, multi-interference, strong coupling, strong nonlinearity and large time delay process. Actual production experience shows that three factors (the temperature in central tower, the temperature difference in the middle of the tower and in the upper of tower, the liquid height) influence the reaction process [15], [16].

An important indicator of carbonization process is sodium bicarbonate crystallization particle size. To obtain larger crystalline particles, the solution supersaturated must be controlled. The nucleation temperature, which is called the critical temperature, is usually in the middle of carbonation tower. Therefore, the temperature in the middle of tower directly affects the quality of crystalloid. The temperature in the middle of tower is controlled by the temperature in lower part and air flows. Actually, the temperature in lower part is set to a fixed value; air flow rate is changed to control the reaction.

The quality of crystalloid is related with the overall temperature distribution in the tower. In the actual production, the temperature difference between the middle of the tower and the upper of tower can reflect the overall temperature distribution in the tower. It is controlled by changing the air flow rate, which changes the ratio of the air flow rate in the middle part and in lower part. Thereby the temperature difference in the middle of the tower and in the upper of tower is stabilized.

The quality of crystalloid is related with the liquid height. If the liquid height level is too high, tail gas is mixed with the liquid. In addition, the tower pressure and the quantity of soda increase during temperature rising. The quality of crystalloid and the alkali elements are affected. The liquid height level is measured difficultly, so the tower pressure is instead.

Through analysis, the tower temperature is controlled by air flow rate in lower part. We use RBF neural network to build system model and control. The temperature in central tower, the temperature difference between the middle of the tower and the upper of tower, and the liquid height are regarded as network inputs. The air flow rate in lower part is regarded as network output. The data is normalized to [-1, 1]. A three layers RBF neural network (3 inputs and 1 output) is created. Gaussian function is selected as activation function.

6 Simulation and Analysis

Fuzzy neural network control method has been applied to practical production process, and obtained some control effect. But the synthetic ammonia decarburization industrial process is a complicated process with nonlinearity, large delay, strong coupling. Fuzzy neural network control method has disadvantage (adjustment error is big). The result is not satisfied. In this paper, we bring up a RBF neural network based on nonlinear particle filter and do some simulation experiments. The result is compared to Fuzzy neural network control method.

One month data was collected in the synthetic ammonia decarburization production process of a co-alkali firm. 6000 samples were selected to train the RBF neural network. When the model is established, the weights of RBF neural network are optimized by the nonlinear particle filter algorithm. Choose 30000 as the particle number. The RBF neural network is packaged as simulink module. An initial Variation of factors is set. The result of the simulation is the response of the system.

PF-RBFNN (particle filter-RBF neural network) and Fuzzy NN (fuzzy neural network) are applied to the synthetic ammonia decarburization. The results are shown in Fig. 3 and Fig. 4. Compared with the experiment results, it is found that the control effect of PF-RBFNN is better.

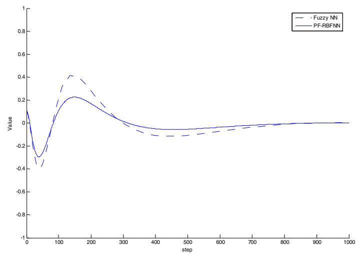


Fig. 3. The temperature transformation in the middle part of tower

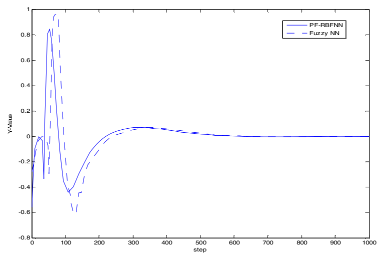


Fig. 4. Control of lower air flow changes in the carbonation tower

In Fig. 3, the result in static and dynamic performance of PF-RBFNN control method is better than Fuzzy NN control method. System response time is shorter; fluctuate range of the temperature in central tower is smaller. From Fig. 4, lower carbonation tower fluctuation in air flow decreases, and the control time is shorter. Simulation results shows that control accuracy and dynamic parameters are significantly increased.

7 Conclusion

The synthetic ammonia decarburization industrial process is a typical complex industrial process with multi-phases, multi-variables, multi-interference, strong coupling, nonlinear, large delay, etc. In response to this feature, we propose nonlinear non-Gaussian particle filter algorithm for RBF neural network weights optimization, and applied to the ammonia decarburization. Through simulation of the carbonization process, this method has a good performance in complex industrial processes control and provides an effective way to solve a class of complex industrial process modeling and optimization control.

Acknowledgments

This paper is supported by the Natural Science Foundation of Hebei Province (Grant No. F2009000728).

References

1. Li, Y.W., Li, W., Yu, G.Q., Guo, P., Wang, Z.Y.: Applied Research in Fuzzy Neural Network Predictive Control. In: 6th IEEE International Conference on Cognitive Informatics, Lake Tahoe, CA, pp. 408–410 (2007)
2. Hu, S.Q., Jing, Z.L.: Overview of particle filter algorithm. Control and Decision 20(4), 361–365 (2005)
3. Gordon, N., Salmond, D.: Novel approach to non-linear and non-Gaussian Bayesian state estimation. Proceedings of Institute Electric Engineering 140, 107–113 (1993)

4. Bolic, M., Hong, S., Djuric, P.M.: Performance and Complexity Analysis of Adaptive Particle Filtering for Tracking Applications. In: International Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, vol. 1, pp. 853–857 (2002)
5. Morales, R., Poole, D.: Estimation and Control of Industrial Processes with Particle Filters. Department of Computer Science University of British Columbia, Canada, Technical Report, 1–10 (2002)
6. Arulampalam, M.S., Maskell, S., Gordon, N., Clapp, T.: A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. IEEE Transactions on Signal Processing 50(2), 174–188 (2002)
7. Liu, G.C., Wang, Y.J.: Visual tracking by particle filtering in a dynamic environment. International Journal of Modelling, Identification and Control 10(1-2), 72–80 (2010)
8. Nahas, E.P., Henson, M.A., Seborg, D.E.: Nonlinear internal model control strategy for neural network models. Computers Chem. Eng. 16(12), 1039–1057 (1992)
9. Carpenter, J., Clifford, P.: Improved particle filter for nonlinear problems. IEE Proceedings of Radar, Sonar and Navigation 1, 2–7 (1999)
10. Xia, H.: A fast identification algorithm for box-cox transformation based radial basis function neural network. IEEE Transactions on Neural Networks 17(4), 1064–1069 (2006)
11. Chen, S., Cowan, C., Grant, P.: Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks. IEEE International on Neural Networks 2(2), 302–309 (1991)
12. Bai, E.W., Fotedar, S., Moy, A.: Modelling and Parameter Estimation of a cell system. Int. J. of Modelling, Identification and Control 6(1), 72–80 (2009)
13. Zhao, H.C., Gu, W.J.: RBF neural network-based sliding mode control for a ballistic missile. Int. J. of Modelling, Identification and Control 8(2), 107–113 (2009)
14. Chen, Y.P., Wang, X.L., Huang, S.T.: Neural network learning algorithm based on particle filter. Engineering Journal of Wuhan University 28(6), 86–88 (2006)
15. Ran, H.C., Li, Y.W., Xue, Z.T.: The Automatic Control System for the Process of the Synthetic Ammonia Decarbornization. Journal of Hebei University of Science and Technology 24(1), 43–47 (2003)
16. Li, T.F., Feng, G.L., Zhong, B.X., Liu, Y.C.: Analysis on Control Strategy for a Kind of Uncertain Complexity System. Journal of Chongqing University (Natural Science Edition) 26(1), 46–47 (2003)

Elman-Style Process Neural Network with Application to Aircraft Engine Health Condition Monitoring

Gang Ding and Lin Lin

School of Mechatronics Engineering, Harbin Institute of Technology,
Harbin 150001, P.R. China
dingganghit@163.com

Abstract. In reality, the inputs of many complicated systems are continuous time-varying functions. It is difficult for traditional Elman neural networks (ENN) to simulate such complicated nonlinear systems directly because their inputs are all instantaneous constant values. To overcome this limitation, an Elman-style process neural network (EPNN) is proposed in this paper. From the point view of architecture, the EPNN is similar to the ENN. The major characteristics which distinguish the EPNN from the ENN lie in the fact that the inputs and the connection weights of the EPNN are time-varying functions. A corresponding learning algorithm based on the expansion of the orthogonal basis functions is developed. The effectiveness of the EPNN and its learning algorithm is proved by the lubricating oil iron concentration prediction in the aircraft engine health condition monitoring, and the application test results also indicate that the EPNN has a faster learning speed and a higher accuracy than the same scale ENN.

Keywords: Elman-style process neural network; Orthogonal basis function; Time series prediction; Aircraft engine health condition monitoring.

1 Introduction

Artificial neuron models are extremely simple abstraction of biological neurons. Since McCulloch and Pitts proposed MP neuron model in 1943 [1], artificial neural networks have received significant attention due to their capability of self-learning and self-adapting. In 1989, Hornik and Funahashi proved that multilayer feedforward neural networks can approximate any continuous function [2, 3]. It seems that artificial neural networks have great potential to high nonlinear and uncertain systems.

Unfortunately, an inherent disadvantage of these artificial neural networks is that their inputs are all instantaneous constant values. However, actually physiological researches indicate that, in biological neurons, the states of the synapses are interrelated with relative time of the input impulse. It makes synapse build up in a 20ms time window, and makes synapse restrain in another 20ms time window. In another word, there exists a reciprocity process between two connected neurons and this process lasts for 40ms [4]. At the same time, in practical engineering, the inputs of many complicated systems are time-varying functions or processes. In order to solve these problems, He and Liang proposed process neuron model in 2000 [5].

© Springer-Verlag Berlin Heidelberg 2011

From a point view of architecture, the process neuron is similar to the traditional artificial neuron. The major difference is that the inputs and the corresponding connection weights of the process neuron can be time-varying functions.

Process neural networks are parallel computational models comprised of densely interconnected process neurons [6, 7]. A particularly important element of the design of the process neural networks is the choice of the architecture. Generally, multilayer feedforward neural network architecture is widely adopted. However, it has been shown that feedback occurs in almost every part of the nervous systems of every animal [8]. In response to this physiological phenomenon, a number of recurrent architectures have been proposed [9, 10]. Perhaps the most widely used, at present, is the Elman neural network (ENN) [11, 12].

In this paper, an Elman-style process neural network (EPNN) model is proposed to aim at solving the complicated problems in real systems where the inputs are time-varying functions. The EPNN has a feedback connection from the output of the hidden layer to the input of the hidden layer like the traditional ENN. The major characteristics which distinguish the EPNN from the ENN lie in the fact that the inputs and the connection weights of the EPNN are time-varying functions. The EPNN proposed in this paper consists of four layers: an input layer with nodes, a hidden layer with process neurons, a context layer with process neurons and an output layer with process neurons. The EPNN is subjected to the constraint that. A corresponding learning algorithm is developed. In order to simplify the computational complexity of the learning algorithm, a group of appropriate orthogonal basis functions are introduced into the input space of the EPNN to expand the input functions and the connection weight functions. The effectiveness of the EPNN and its learning algorithm is validated by the lubricating oil iron concentration prediction in the aircraft engine condition monitoring, and the comparative application test results highlight the performance of the EPNN.

The plan of this paper is as follows: In section 2, the process neuron model is reviewed, which sets the foundation for the remainder of this paper. In section 3, the topological structure of the EPNN is described. In section 4, a learning algorithm based on the expansion of the orthogonal basis functions for the EPNN is developed. In section 5, the effectiveness of the EPNN and its learning algorithm is proved by the lubricating oil iron concentration prediction in the aircraft engine condition monitoring. Conclusions are given in section 6.

2 Process Neuron Model

The process neuron model is composed of three sections: inputs, a processing operator and output. This is based on the fact that a biological neuron is composed of three basic parts: a dendrite, a soma and an axonal tree. Generally, the inputs and the connection weights of the neuron in a traditional neural network are discrete constant values. However, the inputs and the connection weights of the process neuron are continuous time-varying functions. An aggregation operator on time is added to the process neuron, which provides the process neuron with the capability of handing simultaneously two items of dimension information of time and space. The process neuron architecture is depicted in Fig. (1).

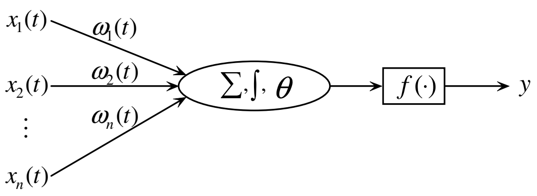


Fig. 1. The sketch diagram of process neuron model

The output of the process neuron model can be expressed as

$$y = f(\sum_{i=1}^{n} \int_0^T \omega_i(t) x_i(t) dt - \theta). \tag{1}$$

Where $x_i(t) \in C[0,T]$ is the i-th input function, $\omega_i(t)$ is the i-th weight function, θ is the threshold, and $f(\cdot)$ is the activation function.

3 Elman-Style Process Neural Network Model

The topological structure of the EPNN model proposed in this paper is illustrated in Fig. (2).

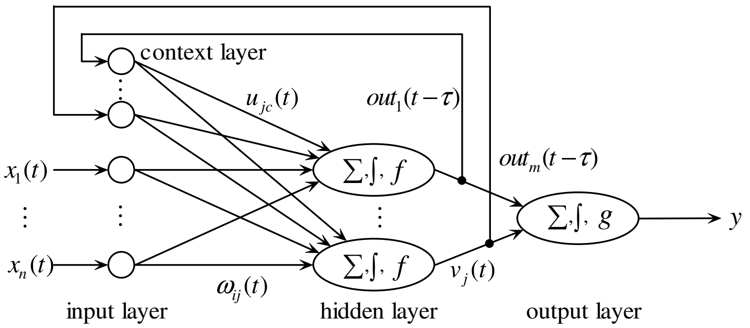


Fig. 2. The topological structure of the EPNN model

The EPNN presented in this section is composed of four layers, with the addition of a feedback connection from the output of the hidden layer to the input of the hidden layer, i.e. with a feedback connection from the context layer to the hidden layer. The input layer is comprised of n nodes. The hidden layer is comprised of m process neurons. The context layer is also comprised of m process neurons. The last layer is output layer, to reduce the complexity, we only consider the case of one output process neuron. Every input node is connected to every process neuron in the hidden layer, as is every process neuron in the context layer. Similarly, there are massively parallel connections between the hidden layer and the output layer. Both the inputs of the EPNN and the process neurons in the context layer activate the process neurons in the hidden layer; and then the process neurons in the hidden layer feed forward to activate the process neuron in the output layer. The process neurons in the hidden layer also feed back to activate the process neurons in the context layer.

The inputs and the corresponding connection weights of the EPNN are continuous time-varying functions. The input function of the EPNN is

$$X(t) = (x_1(t), \cdots, x_n(t)) . \tag{2}$$

The input of the hidden layer can be expressed as

$$net_j(t) = \sum_{i=1}^{n} \omega_{ij}(t) x_i(t) + \sum_{c=1}^{m} u_{jc}(t) out_c(t-\tau) . \tag{3}$$

Where $net_j(t)$ $(j = 1, \cdots, m)$ denotes the input of the j-th process neuron in the hidden layer at time t . $\omega_{ij}(t)$ is the connection weight function between the j-th process neuron in the hidden layer and the i-th node in the input layer. $out_c(t-\tau)$ $(c = 1, \cdots, m)$ denotes the output of the c-th process neuron in the hidden layer at time $t-\tau$, where τ is the time delay, and it also denotes the input of the c-th process neuron in the context layer at time t . $u_{jc}(t)$ is the connection weight function between the j-th process neuron in the hidden layer and the c-th process neuron in the context layer.

The output of the hidden layer can be expressed as

$$out_j(t) = f(net_j(t) - \theta_j^{(1)}) . \tag{4}$$

Where $out_j(t)$ denotes the output of the j-th process neuron in the hidden layer at time t . $\theta_j^{(1)}$ is the threshold of the j-th process neuron in the hidden layer. $f(\cdot)$ is the activation function of the process neurons in the hidden layer, it is assumed to be a differentiable nonlinear function.

Suppose that the activation function $g(\cdot)$ of the process neuron in the output layer is a linear function. Then, the output of the EPNN model can be written as

$$y = \sum_{j=1}^{m} \int_0^T v_j(t) out_j(t) dt - \theta^{(2)} . \tag{5}$$

Where y denotes the output of the EPNN. $v_j(t)$ is the connection weight function between the j-th process neuron in the hidden layer and the process neuron in the output layer, and $t \in [0, T]$. $\theta^{(2)}$ is the threshold of the process neuron in the output layer.

4 Learning Algorithm

In this section, it is to be demonstrated that the proposed EPNN model does not require a complex learning algorithm, and that it is possible to utilize a simple gradient descent learning algorithm to train the network.

4.1 Basic Learning Algorithm

Consider a set of S input/output pairs $\{X^s(t), d^s\}$, where $s = 1, \cdots, S$, d^s represents the desired network output upon presentation of $X^s(t)$. Suppose that y^s is the actual output corresponding to d^s. Thus, the mean square error of the EPNN model can be defined below

$$E = \frac{1}{2}\sum_{s=1}^{S}(y^s - d^s)^2 . \tag{6}$$

The learning algorithm for the EPNN should adjust the network parameters such as $\omega_{ij}(t)$, $u_{jc}(t)$, $v_j(t)$, $\theta_j^{(1)}$, and $\theta^{(2)}$ to minimize the mean square error E of the EPNN. Applying gradient descent method on this estimate of the mean square error, the delta learning rules can be defined as follows

$$v_j(t) = v_j(t) + \alpha\Delta v_j(t) . \tag{7}$$

$$u_{jc}(t) = u_{jc}(t) + \beta\Delta u_{jc}(t) . \tag{8}$$

$$\omega_{ij}(t) = \omega_{ij}(t) + \gamma\Delta\omega_{ij}(t) . \tag{9}$$

$$\theta_j^{(1)} = \theta_j^{(1)} + \lambda\Delta\theta_j^{(1)} . \tag{10}$$

$$\theta^{(2)} = \theta^{(2)} + \eta\Delta\theta^{(2)} . \tag{11}$$

Where $\alpha, \beta, \gamma, \lambda, \eta$ are the corresponding learning rates.

According to Equation (6), we have

$$\Delta v_j(t) = -\frac{\partial E}{\partial v_j(t)} = -\sum_{s=1}^{S}(y^s - d^s)\frac{\partial y^s}{\partial v_j(t)} . \tag{12}$$

$$\Delta u_{jc}(t) = -\frac{\partial E}{\partial u_{jc}(t)} = -\sum_{s=1}^{S}(y^s - d^s)\frac{\partial y^s}{\partial u_{jc}(t)} . \tag{13}$$

$$\Delta\omega_{ij}(t) = -\frac{\partial E}{\partial\omega_{ij}(t)} = -\sum_{s=1}^{S}(y^s - d^s)\frac{\partial y^s}{\partial\omega_{ij}(t)} . \tag{14}$$

$$\Delta\theta_j^{(1)} = -\frac{\partial E}{\partial\theta_j^{(1)}} = -\sum_{s=1}^{S}(y^s - d^s)\frac{\partial y^s}{\partial\theta_j^{(1)}} . \tag{15}$$

$$\Delta\theta^{(2)} = -\frac{\partial E}{\partial\theta^{(2)}} = \sum_{s=1}^{S}(y^s - d^s) . \tag{16}$$

It is obvious that Equation (12)~(16) are very difficult to be implemented directly. In order to raise the efficiency of the computation and the adaptability to practical problems solving of the EPNN, a group of appropriate orthogonal basis functions are introduced into the input function space of the EPNN.

4.2 Learning Algorithm Based on Orthogonal Basis Function

We know $x_i(t) \in C[0,T]$, by Weierstrass approximation theorem, if any $\varepsilon > 0$ is given, then there exists a polynomial $P(t)$ on $[0,T]$ such that $|x_i(t) - P(t)| < \varepsilon$ for all $t \in [0,T]$. In short, any $x_i(t) \in C[0,T]$ can be uniformly approximated on $[0,T]$ by polynomials $P_l(t)(l = 1,\cdots,L)$ to any degree of accuracy. Therefore, $x_i(t)$ can be written in the form $x_i(t) = \sum_{l=1}^{L} c_l P_l(t)$, where the coefficient $c_l \in R$ is easy to find. Suppose that the set of $P_l(t)$ is independent. There is a relationship between orthogonality and independence. It is possible to convert a set of independent functions $P_l(t)$ into a set of orthogonal functions $b_l(t)$ that spans the same space. The standard procedure to accomplish this conversion is called Gram-Schmidt orthogonalization. Therefore, the input function $x_i(t)$, weight functions $v_j(t)$, $u_{jc}(t)$ and $\omega_{ij}(t)$ can be expanded at the same time respectively as follows

$$x_i(t) = \sum_{l=1}^{L} a_{il} b_l(t) . \tag{17}$$

$$v_j(t) = \sum_{l=1}^{L} v_{jl} b_l(t) . \tag{18}$$

$$u_{jc}(t) = \sum_{l=1}^{L} u_{jcl} b_l(t) . \tag{19}$$

$$\omega_{ij}(t) = \sum_{l=1}^{L} \omega_{ijl} b_l(t) . \tag{20}$$

where $a_{il}, v_{jl}, u_{jcl}, \omega_{ijl} \in R$ are the corresponding coefficients.

Assume that $[0,T]$ contains K equally spaced nodes t_k, where $t_k = t_0 + kh$, $k = 1,\cdots,K$, $t_0 = 0$ and $h = T/K$. Thus, Equation (17)~(20) can be written in the following forms

$$x_i(t_k) = \sum_{l=1}^{L} a_{il} b_l(t_k) . \tag{21}$$

$$v_j(t_k) = \sum_{l=1}^{L} v_{jl} b_l(t_k) . \tag{22}$$

$$u_{jc}(t_k) = \sum_{l=1}^{L} u_{jcl} b_l(t_k) . \tag{23}$$

$$\omega_{ij}(t_k) = \sum_{l=1}^{L} \omega_{ijl} b_l(t_k) . \tag{24}$$

Then, $out_j(t_k)$ can be written as

$$out_j(t_k) = f\left(\sum_{i=1}^{n}[(\sum_{l=1}^{L}\omega_{ijl} b_l(t_k))(\sum_{l=1}^{L} a_{il} b_l(t_k))] + \sum_{c=1}^{m}[(\sum_{l=1}^{L} u_{jcl} b_l(t_k)) out_c(t_{k-1})] - \theta_j^{(1)}\right) . \tag{25}$$

Thus, Equation (5) can be substituted by

$$y=\sum_{j=1}^{m}\sum_{k=1}^{K}[(\sum_{l=1}^{L}v_{jl}b_l(t_k))out_j(t_k)h]-\theta^{(2)} . \tag{26}$$

This allows us to conveniently write out an expression for mean square error of the EPNN model as

$$E=\frac{1}{2}\sum_{j=1}^{m}\sum_{k=1}^{K}[(\sum_{l=1}^{L}v_{jl}b_l(t_k))out_j^s(t_k)h]-\theta^{(2)}-d^s)^2 . \tag{27}$$

The learning algorithm will update the network parameters v_{jl} , u_{jcl} , ω_{ijl} , $\theta_j^{(1)}$ and $\theta^{(2)}$ to minimize the mean square error of the EPNN model according to Equation (27).

For the convenience of analysis, Q^s is defined below

$$Q^s=\sum_{i=1}^{n}[(\sum_{l=1}^{L}\omega_{ijl}b_l(t_k))(\sum_{l=1}^{L}a_{il}^s b_l(t_k))]+\sum_{c=1}^{m}[(\sum_{l=1}^{L}u_{jcl}b_l(t_k))out_c^s(t_{k-1})]-\theta_j^{(1)} .$$

Applying the gradient descent method on Equation (27), we have

$$\Delta v_{jl}=-\frac{\partial E}{\partial v_{jl}}=-\sum_{s=1}^{S}(y^s-d^s)\sum_{k=1}^{K}b_l(t_k)f(Q^s)h . \tag{28}$$

$$\Delta u_{jcl}=-\frac{\partial E}{\partial u_{jcl}}=-\sum_{s=1}^{S}(y^s-d^s)\sum_{k=1}^{K}v_{jl}b_l(t_k)f'(Q^s)hb_l(t_k)out_c^s(t_{k-1}) . \tag{29}$$

$$\Delta \omega_{ijl}=-\frac{\partial E}{\partial \omega_{ijl}}=-\sum_{s=1}^{S}(y^s-d^s)\sum_{k=1}^{K}v_{jl}b_l(t_k)f'(Q^s)hb_l(t_k)(\sum_{l=1}^{L}a_{il}^s b_l(t_k)) . \tag{30}$$

$$\Delta \theta_j^{(1)}=-\frac{\partial E}{\partial \theta_j^{(1)}}=\sum_{s=1}^{S}(y^s-d^s)\sum_{k=1}^{K}v_{jl}b_l(t_k)f'(Q^s)h . \tag{31}$$

$$\Delta \theta^{(2)}=-\frac{\partial E}{\partial \theta^{(2)}}=\sum_{s=1}^{S}(y^s-d^s) . \tag{32}$$

This yields the update rules for weighs and thresholds for each iteration step to be

$$v_{jl}=v_{jl}+\alpha\Delta v_{jl} . \tag{33}$$

$$u_{jcl}=u_{jcl}+\beta\Delta u_{jcl} . \tag{34}$$

$$\omega_{ijl}=\omega_{ijl}+\gamma\Delta\omega_{ijl} . \tag{35}$$

$$\theta_j^{(1)}=\theta_j^{(1)}+\lambda\Delta\theta_j^{(1)} . \tag{36}$$

$$\theta^{(2)}=\theta^{(2)}+\eta\Delta\theta^{(2)} . \tag{37}$$

Where $\alpha,\beta,\gamma,\lambda,\eta$ are the corresponding learning rates.

It is usually possible to express the derivative of the activation $f(\cdot)$ in terms of itself. For example, for the logistic function $f(z)=\frac{1}{1+e^{-z}}$, $f'(z)=f(z)f(1-f(z))$.

The iterations of the learning algorithm based on the orthogonal basis functions for the EPNN can be summarized as follows:

step 1 Select appropriate orthogonal basis functions to expand the input functions and the corresponding weight functions.

step 2 Initialize v_{jl}, u_{jcl}, ω_{ijl}, $\theta_j^{(1)}$ and $\theta^{(2)}$.

step 3 Set output error goal $\varepsilon > 0$; initialize iteration $k = 0$, and set max iteration as K.

step 4 Utilize Equation (27) to calculate mean square error E, if $E < \varepsilon$ or $k > K$, go to step6; otherwise, go to step5.

step 5 Update v_{jl}, u_{jcl}, ω_{ijl}, $\theta_j^{(1)}$ and $\theta^{(2)}$ according to Equation (33)~(37); $s+1 \to s$, go to step4.

step 6 Output results; stop.

5 Application Test

In this section, the EPNN proposed in this paper is utilized to predict the lubricating oil iron concentration in the aircraft engine condition monitoring.

Aircraft engine is a complicated nonlinear system, which operates under high temperature and speed conditions. Operating such a modern technical system, calls for good maintenance to keep the system in an optimal operational condition. Predictive maintenance is preferred over scheduled maintenance. The scheduled maintenance follows a set of schedule: after a specified usage period, the engines are disassembled and overhauled, irrespective of their health condition. Such scheduled maintenance is costly. Nowadays, predictive maintenance is becoming more widely adopted. The engines are removed and examined or repaired only when some faults occur. The predictive maintenance method improved both the economics and reliability of the operation.

The lubrication system is an important working system of the aircraft engine. The lubricating oil monitoring is essential in terms of the flight safety and also for reduction of the predictive maintenance cost. The monitoring analysis of the lubricating oil taken from the aircraft engine gives an indication of its suitability for continued use and provides important information about the health condition of the lubricated components within the aircraft engine. The concentration of mental elements in the lubricating oil contains a great deal of information of the aircraft engine's health condition. By predicting the tendency of the mental elements concentration in the lubricating oil, maintenance engineers can judge the wear abrasion of the lubricated components in the aircraft engine and can deduce the mechanical faults of the aircraft engine in advance. However, the lubricating oil is influenced by many complicated factors and varying continuously with time. It is difficult or impossible to predict the tendency of the mental elements concentration in the lubricating oil by a determinate mathematic model. In this paper, the prediction of the lubricating oil iron concentration by the EPNN is presented.

The data used in this paper was taken from some aircraft, and the sampling interval is about 24 hours. A time series of the lubricating oil iron (Fe) concentration with 155 discrete points such as $\{Fe_i\}_{i=1}^{155}$ is depicted in Fig. (3).

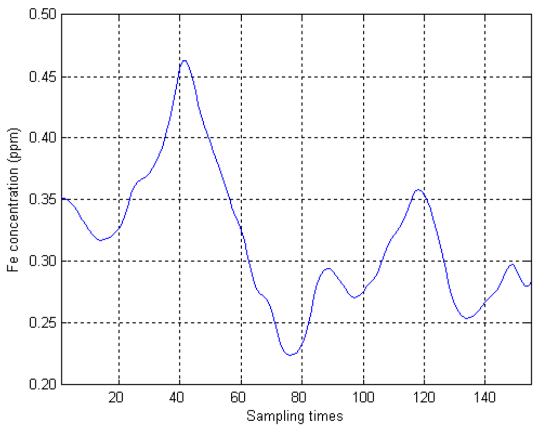


Fig. 3. Fe concentration time series

$(Fe_i, Fe_{i+1}, \cdots, Fe_{i+4})$ is used to generate an input function IF_i by the nonlinear least-squares method, where $i = 1, \cdots, 150$, and Fe_{i+5} is used as the desired output of the EPNN corresponding to IF_i. Thus, we get 150 couples of samples such as $\{IF_i, Fe_{i+5}\}_{i=1}^{150}$. The samples $\{IF_i, Fe_{i+5}\}_{i=1}^{100}$ are selected to train the EPNN. The EPNN model used in this section is composed of 4 layers. The input layer has one node, the hidden layer is consisted of 5 process neurons, the context layer is also consisted of 5 process neurons, the last layer is output layer with one process neuron. The orthogonal Legendre basis functions are selected to expand the input functions and the corresponding connection weight functions. The error goal is set to 0.0001, and the learning rate is set to 0.01, the max iteration number is set to 5000. After 1667 iterations, the EPNN has converged.

The samples $\{IF_i, Fe_{i+5}\}_{i=101}^{150}$ are selected to test the EPNN. The test results as shown in Fig. 5 indicate that the EPNN seems to perform well and appears suitable for using as a predictive maintenance tool.

In order to compare with the traditional ENN, an ENN model with the same scale of the EPNN used in this section is trained by the same lubricating oil iron concentration data. Five consecutive points in the lubricating oil iron concentration time series are used as inputs and the next point is used as the corresponding desired output of the ENN. Thus, we also get 150 couples of samples such as $\{(Fe_i, Fe_{i+1}, \cdots, Fe_{i+4}), Fe_{i+5}\}_{i=1}^{150}$. The first 100 samples such as $\{(Fe_i, Fe_{i+1}, \cdots, Fe_{i+4}), Fe_{i+5}\}_{i=1}^{100}$ are used to train the ENN, and the next 50 samples such as $\{(Fe_i, Fe_{i+1}, \cdots, Fe_{i+4}), Fe_{i+5}\}_{i=101}^{150}$ are selected to test the EN. After 3101 iterations, the ENN model has converged. The ENN is tested by the testing samples, and the test results are also depicted in Fig. (4).

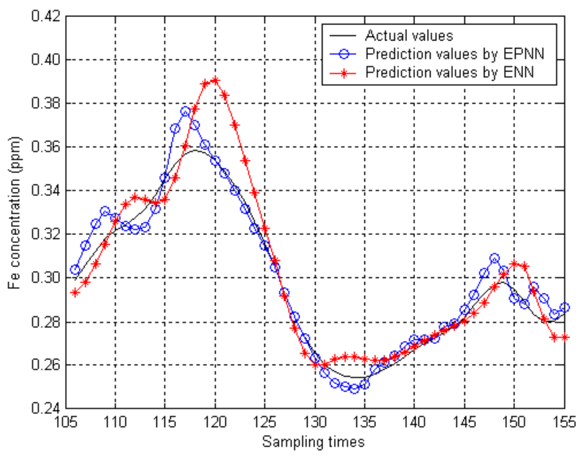


Fig. 4. Fe concentration time series prediction

The comparative test results as shown in Fig. 5 indicate that the EPNN is more effective than the ENN. In fact, the ENN is only a special case of the EPNN. Since the inputs of the EPNN are time-varying functions, the EPNN can expand the application domains of the traditional ENN. It can be seen from Fig. 5 that the EPNN has a faster convergence speed and a higher accuracy than the ENN. This suggests that fairly complex problems in the practical engineering can be tackled by the EPNN.

6 Conclusion

In order to solve the complex problems in real systems where the inputs are time-varying functions, an EPNN model is proposed in this paper, which has a typical architecture like the traditional ENN model. The major difference between the EPNN and the ENN is that the inputs and the corresponding connection weights of the EPNN are not discrete constant values but continuous time-varying functions. A learning algorithm for the EPNN model is developed. In consideration of the complexity of the aggregation operation of time in the EPNN, a group of appropriate orthogonal functions in the input function space of the EPNN are selected, and then the input functions and the corresponding connection weight functions are expanded by the same orthogonal basis functions. With the orthogonality of the orthogonal basis functions, the aggregation operation of process neurons to time is simplified. This application shows that the learning algorithm based on the expansion of the orthogonal basis functions simplifies the computing complexity of the EPNN, and raise the efficiency of the network learning and the adaptability to real problem resolving. The effectiveness of the EPNN and its learning algorithm is proved by the lubricating oil iron concentration prediction in the aircraft engine condition monitoring, and the application test results indicate that the EPNN has two advantages. Firstly, since its inputs are time-varying functions, the EPNN can expand the application domains of the traditional ENN. Secondly, compared with the ENN model, the EPNN model has a faster convergence speed and a higher accuracy.

Acknowledgments. This study was supported by the National Natural Science Foundation of China under Grant No. 50805032.

References

1. McCullon, W., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics 5, 115–133 (1943)
2. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
3. Funahashi, K.: On the approximate realization of continuous mappings by neural networks. Neural Networks 2, 183–192 (1989)
4. Zhang, L.I., Tao, H.W., Holt, C.E.: A critical window for cooperation and competition among developing retinotectal synapses. Nature 395, 37–44 (1998)
5. He, X.G., Liang, J.Z.: Some theoretical issues on procedure neural networks. Engineering Science 2, 40–44 (2000)
6. Ding, G., Lin, L., Zhong, S.S.: Functional time series prediction using process neural network. Chinese Physics Letters 26, 090502-1–090502-4 (2009)
7. Ding, G., Zhong, S.S.: Time series prediction using wavelet process neural network. Chinese Physics B 17, 1998–2003 (2008)
8. Freeman, W.J.: Mass action in the nervous system, New York (1975)
9. Rajan, K., Abbott, L.F., Sompolinsky, H.: Stimulus-dependent suppression of chaos in recurrent neural networks. Physical Review E 82, 011903 (2010)
10. Mirikitani, D.T., Nikolaev, N.: Recursive bayesian recurrent neural networks for time-series modeling. IEEE Transactions on Neural Networks 21, 262–274 (2010)
11. Elman, J.L.: Finding structure in time. Cognitive Science 14, 179–211 (1990)
12. Portegys, T.E.: A maze learning comparison of Elman, long short-term memory, and Mona neural networks. Neural Networks 23, 306–313 (2010)

A Novel Spatial Architecture Artificial Neural Network Based on Multilayer Feedforward Network with Mutual Inhibition among Hidden Units

Gang Yang[1], Junfei Qiao[1], and Mingzhe Yuan[2]

[1] Intelligent Systems Institute, College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing, 100124, China
[2] Shenyang Institute of Automation Chinese Academy of Science, Shenyang, 110016, China
hank.yang2010@gmail.com, isibox@sina.com

Abstract. We propose a Spatial Artificial Neural Network (SANN) with spatial architecture which consists of a multilayer feedforward neural network with hidden units adopt recurrent lateral inhibition connection, all input and hidden neurons have synapses connections with the output neurons. In addition, a supervised learning algorithm based on error back propagation is developed. The proposed network has shown a superior generalization capability in simulations with pattern recognition and non-linear function approximation problems. And, the experimental also shown that SANN has the capability of avoiding local minima problem.

Keywords: multilayer feedforward network, spatial span connection, lateral inhibition mechanism.

1 Introduction

Warren McCulloch and Walter Pitts [1] gave an academic background of applying artificial neural networks (ANNs) because they have shown that ANNs have the capability of computing any arithmetic or logical function in principle. Because of the characteristic of resembling the process we go through learning something [2], good at providing fast and close approximations of the correct answer, and best at identifying patterns and trends in data, ANNs have already been successfully applied in many industries and daily life for prediction and forecasting [3-8].

Theoretical research of ANNs major focus on two points: architecture (BP, Hopfield network, RBF etc.) and learning algorithm (Back propagation, Levenberg-Marquardt, Conjugate Gradient algorithm etc.). These structures of ANNs basically are planar and symmetrical connection. But the animal's neuronal system is there dimensional and asymmetry; and the neurons of brain in previous layers have narrow perception field but have wider perception fields in deep layers. On the other hand, structure defines function in ANNs. So, there

© Springer-Verlag Berlin Heidelberg 2011

are some structure studies, for instance, based on functional expansion, Chao proposed a new pyramid network [9] and analyzed the representation and generalization theoretically and experimentally; Jaeger presented Echo State Networks [10], which combined the trainable linear combination of nonlinear response signals of internal units and the desired output signals. Research of cortex shown that the inhibition of cerebral cortex may avoid the excitatory activity caused by a stimulus ripple across the entire network and prevent the confused jumble resulted by the overlapping signals [11].

We proposes a neural network with new spatial architecture called Spatial Artificial Neural Network (SANN) to tradeoff the structure and functions and aim to achieve higher representation and generalization capability. SANN is based on a multilayer feed-forward network, spatial span connection and recurrent lateral inhibitory connection mechanism. Through SANN, all non-output neurons have connections to the output neurons. Supervised training rules of SANN are then shown for weights update and learning.

The proposed network applied to several benchmark problems indicated that this network yields significantly representation capability and generalization performance, and it also can avoid the local minima problems in the supervised learning.

2 Definition of Spatial Artificial Neural Network

We define the spatial artificial neural network as follows, and the topology structure is shown in Fig. 1. The first part of SANN architecture is so-called the basic network lies in the center of SANN architecture. Actually the basic network is a multilayer feedforward network which adds mutual inhibition mechanism among the hidden units within a same layer. The feedforward connections among neurons are shown in Fig. 1 using arrowhead line, and its mutual inhibition adopts the recurrent lateral inhibition mechanism, as shown in Fig. 2. The second part of SANN is the spatial span connections part which is shown with arrowhead broken lines.

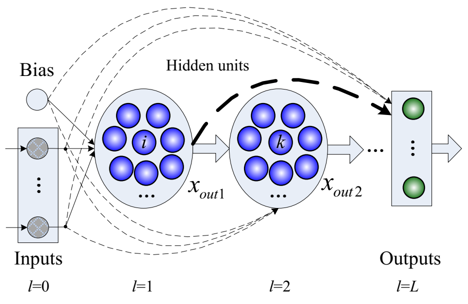


Fig. 1. Architecture of SANN

This basic network is used for transferring information from the environmental inputs to the network outputs. The recurrent lateral inhibitory connection of one hidden layer brings, which is widely found in biological neural systems, connection and competition to the hidden units.

The input signals were transferred to hidden units through the feedforward connections, and the outputs of hidden units corresponding to these input signals are called local output. And in the lateral inhibition theory, it is also called activation level. It will inhibit other hidden units with strength of inhibitory coefficients. The feedforward input combines with the inhibitory input (caused by other hidden units and calculated according to the product of the inhibiting neuron's activation level and the corresponding recurrent lateral inhibitory coefficient) as the total input of hidden units, and the output was transferred to the next layers as the input signal.

The accessorial spatial span connections lay over the basic network of SANN. This part make that any two units of SANN in different layers may have direction connect according to the spatial span connection and weights. The global output of any unit through this span connection transfer signals to the next two or more layer neurons.

Both the inhibitory connection of hidden units and the spatial span connection between different layers guarantee the structure of SANN is a there dimensional and asymmetry system, which resembles the biological neuronal system structure.

3 Mathematic Model of SANN

3.1 Review of Lateral Inhibition

Lateral inhibition (LI) is one type of inhibitions of cerebral cortex. It is one of the basic principles of information treatment within neural system and describes the capacity of an excited neuron to reduce the activity of its neighbor neurons. In the design of our SANN network, the architecture in hidden layers uses the recurrent lateral inhibition to achieve competition.

The topology of lateral inhibitory connection without self-inhibition adapted in SANN is show in Fig. 2, and the mathematic model is given by [12]

$$y_{o,j} = y_{i,j} - \sum_{r=1, r\neq j}^{h} v_{rj}(y_{o,r} - \theta_{rj}), \quad r = 1, 2, ..., h \tag{1}$$

where h is the number of hidden units, $y_{o,j}$ is the output of j-th hidden neuron, $y_{i,j}$ is the environment stimulates of neuron j received, $v_{rj} = v_{jr}$ is the lateral inhibitory coefficient between neuron r and j, θ_{rj} is the inhibiting threshold value of neuron j due to neuron r.

3.2 Mathematic Model of SANN

Suppose the SANN has L+1 layers including the input and output layers, the l-th (l =1, 2, ..., L-1) hidden layer has n_l units. The input layer is denoted as

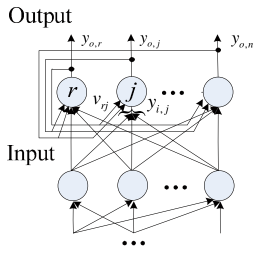


Fig. 2. Recurrent Lateral Inhibition Network

$l = 0$, has n units; and the output layer is denoted as $l = L$, has m units. Symbols i, l and k denote the input, hidden and output units respectively.

The output of the k-th unit in the output layer is given by:

$$o_k = f^L[\sum_{l=1}^{L-1}(\sum_{j=1}^{n_l}\omega_{jk}o_j^l + b_l) + \sum_{i=1}^{n}\omega_{ik}x_i + b_0], \quad k = 1, 2, ..., m \tag{2}$$

where the f^l is the activation function, o_{jl} is the output of the j-th hidden neuron of the l-th layer, it is given by:

$$o_j^l = \begin{cases} f^l[\sum_{i=1}^{n}\omega_{ij}x_i + b_0] - \sum_{r=1}^{n_l} v_{rj}(o_r^{'l} - \theta), & l = 1 \\ f^l[\sum_{q=1}^{l-1}(\sum_{p=1}^{n_q}\omega_{pj}o_p^q + b_q) + \sum_{i=1}^{n}\omega_{ij}x_i + b_0] & \\ - \sum_{r=1}^{n_l} v_{rj}(o_r^{'l} - \theta), & l \in [2, L-1] \end{cases} \tag{3}$$

where v_{rj} is the inhibitory coefficient from neuron r to neuron j, $o_l^{'r}$ is the local output of hidden unit r in l-th layer, namely the activation level of neuron r, it is given by:

$$o_j^{'l} = f^l[\sum_{q=1}^{l-1}(\sum_{p=1}^{n_q}\omega_{pj}o_p^q + b_q) + \sum_{i=1}^{n}\omega_{ij}x_i + b_0], \quad p = 1, 2, ..., n_q \tag{4}$$

From Fig. 1 and Eq. (4) we can find that it is output of the current hidden neuron due to the feedforward input signals, namely the output comes from the input neurons. v_{rj} is the recurrent lateral inhibitory weight between neuron r and j of hidden layer l.

4 Learning Algorithm

Here, we reference the standard back propagation algorithm to derivate the learning algorithm fit for SANN. It is described as follows.

Assume the three-layer SANN with n input neurons, h hidden neurons and one output neurons respectively. The output of the output unit k due to the p-th ($p \in [1, P]$) input sample is given by o_{pk}, whereas the desired output is t_{pk}, and the output of the j-th hidden unit for the p-th input pattern is given by $o_j^{'p}$. Let ω_{jk} be the weight between the k-th output unit and the j-th hidden unit, and ω_{ij} be the weight between the the j-th hidden unit and i-th input unit. The input for the i-th input unit due to the p-th input pattern is denoted by x_{pi}.

According to the above definitions, the output of the j-th neuron in the hidden layer is given by:

$$o_{pj}^{'} = f(\sum_{i=1}^{n} \omega_{ij} x_{pi} + \sum_{r=1}^{h} v_{rj} \bar{o}_{pr}) \tag{5}$$

where, $\bar{o}_{pr} = f(\sum_{i=1}^{n} \omega_{ij} x_{pi})$ is the output of the r-th hidden neuron due to the feedforward input signals, namely the output comes from the input neurons; v_{rj} is the recurrent lateral inhibitory weight between neuron r and j of hidden layer; f is the sigmoid activation function defined as: $f(x) = 1/(1 + e^{-x})$.

Similarly, the output of the k-th unit in the output layer is given by:

$$o_{pk} = f(\sum_{i=1}^{n} \omega_{ik} x_{pi} + \sum_{j=1}^{h} \omega_{jk} o_{pj}^{'}) \tag{6}$$

We define the sum of squared error of output unit with P samples as the system learning target function, to be:

$$E = \frac{1}{2} \sum_{p=1}^{P} (t_{pk} - o_{pk})^2 \tag{7}$$

The learning algorithm based error back propagation is to change the weights iteratively such that the function E Eq. (7) is minimized. The weight updates are proportional to ∂E. According to the chain rule, we can obtain the partial derivative of ∂E with respect to w_{jk} and w_{ik} , and the weight change for the $(t+1)$-th iteration can be expressed as follows,

$$\begin{cases} \Delta\omega_{jk}(t+1) = \mu \sum_{p=1}^{P} \delta_{pk} o_{pj}^{'}, \\ \Delta\omega_{ik}(t+1) = \mu \sum_{p=1}^{P} \delta_{pk} x_{pi}, \ \delta_{pk} = (t_{pk} - o_{pk}) o_{pk} (1 - o_{pk}) \\ \Delta\omega_{ij}(t+1) = \mu \sum_{p=1}^{P} \delta_{pj} x_{pi}, \ \delta_{pj} = o_{pj}^{'} (1 - o_{pj}^{'}) \delta_{pk} \omega_{jk} \end{cases} \tag{8}$$

where the term μ is the learning rate of the gradient method.

It is worthwhile to notice that the above gradient consists two parts, the first term is the basic back propagation in the basic network and the second term is the span back propagation in the spatial span connection network.

The update of SANN weights according to $\omega_{sk}(t+1) = \omega_{sk}(t) + \alpha\Delta\omega_{sk}$. Where the term α is the momentum term used to balance the stability and oscillate which may caused by the value selection of learning rate μ.

5 Numerical Experimental

We use two benchmark problems to compare the performance of SANN using the above supervised learning rule to train and the multilayer feed-forward network is trained by implementation of the back propagation algorithm in Matlab.

In our experiments, 10-fold cross validation is employed on each problem to compare the performance. The result is the average result of the ten folds.

5.1 Classification Capability (XOR Problem)

In order to consider the classification capability of SANN, we chose a simplest problem which is not linearly separable - "exclusive-or" problem which is also discussed in [13-15]. According to the complexity of XOR problem, in this section we consider to train 2-1-1 and 2-2-1 two architectures of SANN on the two dimensional XOR classification problem described in the first two rows of Table 1. Note that the inputs can still be 0 and 1 but the desired values must be changed keeping in mind the signal range.

(1) 2-1-1 SANN
Because only one hidden unit existed in the architecture of 2-1-1 SANN, there is no inhibitory effect or connection between the hidden units.

Set the value of global parameters are as follows: max iteration number $T = 200$, learning rate $\mu = 0.8$, moment term $\alpha = 0.7$, and the error tolerances are (1) $\tau = 1e - 5$; (2) $\tau = 0$ respectively.

The corresponding outputs of two experiments are as shown in the last two rows of Table 1.

Table 1. Real valued XOR patterns and network output

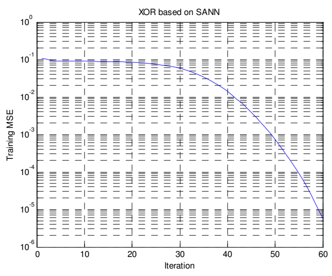
Inputs		Desired	Output1	Output2
0.1	0.1	0.1	0.1013	0.1000
0.1	0.95	0.95	0.9468	0.9500
0.95	0.1	0.95	0.9468	0.9500
0.95	0.95	0.1	0.1048	0.1000

Fig. 3 shows the training errors of the network output and desired output in the supervised algorithm. The test input samples and output as shown in Table 2.

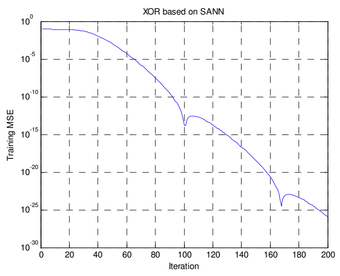
Table 2. Test samples and SANN output

Inputs		Desired	Output1	Output2
0.95	0.95	0.1	0.1048	0.1000
0.1	0.1	0.1	0.1013	0.1000
0.1	0.95	0.95	0.9468	0.9500
0.95	0.1	0.95	0.9468	0.9500

The average result of ten folds for SANN to solve the XOR problem with architecture 2-1-1 of $\tau = 0$ are as follows: running time is 0.0812 s, training mse is 1.77919e-25, test mse is 1.89882e-25.



(1) τ = 1e-5; #60, performance=5.598e-006



(2) τ = 0 #200, performance=1.1724e-026

Fig. 3. The training mse of SANN (2-1-1) network for XOR problem. The architecture is two input neurons, one hidden neurons and one output neurons. $T = 200$, $\mu = 0.8$, $\alpha = 0.7$.

From Table 1-2, and Fig. 3 we can find that a SANN network with only one hidden unit can solve the xor problem at any training accuracy, meanwhile, the similar test accuracy was obtained. Besides, the convergence is very fast that it only needs several iterations.

(2) 2-2-1 BPN and SANN

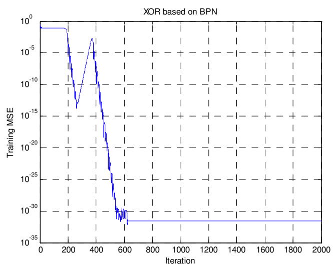
The smallest architecture of multilayer feedforward network is with two hidden units. Here, we compare the generalization capability of SANN and BPN with the architecture of 2-2-1.

Set the value of global parameters are as follows: max iteration number $T = 2000$, learning rate $\mu = 0.8$, moment term $\alpha = 0.7$, and the error tolerances $\tau = 0$. The training mse is shown in Fig. 4, and the average result of ten folds for SANN and BPN respectively is given by in Table 3.

Table 3. The average results of ten folds for BPN and SANN with architecture of 2-2-1

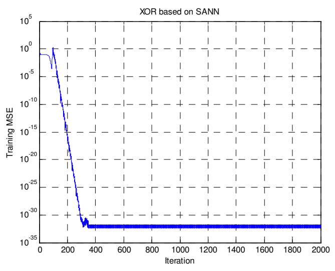
	BP Network	SANN
Global Minimum	3.2741e-33	1.10741e-33
Iteration Number	418	603
Training MSE	3.67811e-31	3.53171e-32
Test MSE	6.5482e-32	4.2695e-32

From this experiment we can find that when they both have two hidden neurons to solve the xor problem, the SANN network has the similar training and test accuracy with BP network, and the convergence of SANN network is faster than BP network, but it need more training time because there are more weights should be calculated than BP network in the learning.



(a) BP Network

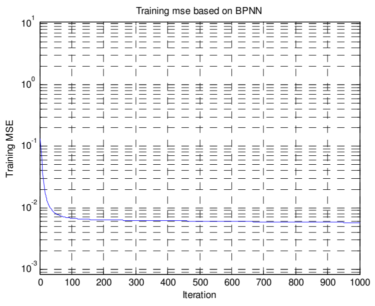
training mse = 2.6385e-032, test mse= 5.5852e-032, min_msel$_{625}$ = 6.3556e-033.



(b) SANN Network

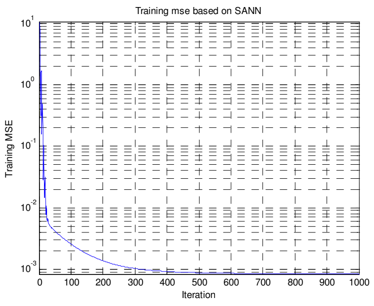
training mse = 1.7141e-032, test mse = 9.629e-033, min_msel$_{349}$ = 4.0445e-033.

Fig. 4. The training mse of SANN network (2-2-1) for XOR problem. The architecture is two input neurons, one hidden neurons and one output neurons. $T = 200$, $\mu = 0.8$, $\alpha = 0.7$, $\tau = 0$.



(1) BP Network (#7)

training mse = 5.740e-03, test mse= 1.04e-02.



(2) SANN Network (#7)

training mse = 8.3094e-04, test mse = 1.8e-03

Fig. 5. The training mse of SANN and BP network (8-4-1) with the parameters of $T = 1000$, $\tau = 1e-05$

5.2 Approximation Capability

Consider the nonlinear function: $f(x) = \frac{x_1x_2+x_3x_4+x_5x_6+x_7x_8}{400}$. Where $x_i(i = 1, 2, ..., 8)$ is drawn at random from [0, 10]. Similar mapping functions have been used in [16-18]. In this paper we use 250 samples to train a SANN network of size 8-4-1 (same with [17]). The parmaters are set as: the learning error tolerance $\tau = 1e-5$, max iteration number $T = 1000$, learning rate $\mu = 1e-4$, moment term $\alpha = 0.7$. The training data set are sampled uniformly in the domain. In order to estimate the generalization capability of SANN network, an independent set of 500 test data is generated at random. The weights are initialized as the

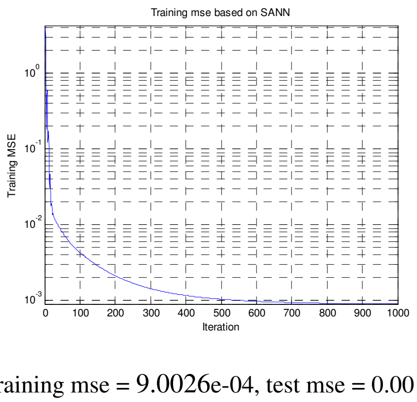


Fig. 6. The training mse of SANN (8-2-1) with the parameters of $T = 1000,\ \tau = 1e{-}05$

random numbers distributed in the interval [-1,1]. As a comprision, a there-layer feed-forward neural network is used and trained using BP algorithm with momentum.

Table 4 lists the comparison results of the SANN network and BP network. The training error results are shown in Fig. 5.

Besides, we also use a SANN network with two hidden units to approximate the nonlinear function, the training error is shown in Fig. 6.

In this experiment, we can find that for a given training accuracy and finite iteration number, SANN network need little hidden units than multilayer feed-forward network. Whether has the same number of hidden units or less than, SANN network performs better than the multilayer feedforward network in terms of both the approximation capability and generalization capability. Besides, SANN can alleviate or avoid the local minima problem.

6 Conclusion and Discussion

In this paper, a novel topology architecture of artificial neural network is proposed based on multilayer feedforward network, lateral inhibition mechanism and spatial span connection mode, and named SANN. The activation function of input neurons is tunable and according to the priori knowledge to chose a proper function as the input neurons' activation function. In this paper, when they connect to the output neurons, choose a nonlinear function, otherwise, choose a linear function. The proposed network shown higher representation and generalization capability in the numerical experiments, and can alleviate or avoid the local minima problem than multilayer feedforward network. But because of the more synapse connections than feedforward network, it needs more time to learn the samples. Design a suitable learning algorithm for SANN is the next useful work.

Acknowledgments. This study is supported by the NSFC (60873043), 863 programs of China (2007AA04Z160), BMSF (4092010), DFME (200800050004), BMET, NCET.

References

1. McCulloch, W., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics 5, 115–433 (1943)
2. Anders, K.: What are Artificial Neural Networks? Nature Biothechnology 26, 195–197 (2008)
3. Stamatis, N., Parthimos, D., Griffith, T.M.: Forecasting chaotic cardiovascular time series with an adaptive slope multilayer perceptron neural network. IEEE Transactions on Biomedical Engineering 46(12), 144–153 (1999)
4. Li, Z., Melek, W.W.: Neural network-based prediction of cardiovascular response due to the gravitational effects. Control and Intelligent Systems 36(1), 81–91 (2008)
5. Liu, D., Ju, C.: Application of an improved BP neural network in business forecasting. In: Sixth World Congress on Intelligent Control and Automation, p. 5 (2006)
6. Pan, W.-T., Lin, W.-Y.: Use probabilistic neural network to construct early warning model for business financial distress. In: 2008 International Conference on Management Science and Engineering (ICMSE), pp. 134–139 (2008)
7. Pan, W.-T.: Use of probabilistic neural network to construct early warning model for business financial distress. Journal of Statistics and Management Systems 11(4), 749–760 (2008)
8. Cheng, M.-Y., Tsai, H.-C., Sudjono, E.: Conceptual cost estimates using evolutionary fuzzy hybrid neural network for projects in construction industry. Expert Systems with Applications 37(6), 4224–4231 (2010)
9. Chao, J., Hoshino, M., Kitamura, T.: A New Pyramid Network and Its Generalization Performance. In: Proc. of IJCNN 2001: International Joint Conference on Neural Networks, Washington D.C., pp. 2811–2816 (2001)
10. Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science 304, 78–80 (2004)
11. Buzsáki, G.: The structure of consciousness. Nature 446, 267 (2007)
12. Sa, L., Kil, R.M.: A Gaussian potential function network with hierarchnically self-organization learning. Neural Networks 4, 207–224 (1991)
13. Simon, W.E., Carter, J.R.: Back propagation learning equations from the minimization of the recursive error. In: IEEE International Conference on Systems Engineering, pp. 155–160 (1989)
14. Jain, L.C.: Hybrid Connectionist Systems in Research and Teaching. IEEE Aerospace and Electronic Systems Magazine 10(3), 14–18 (1995)
15. Li, A.-J., Liu, Y.-H., Luo, S.-W.: On the solution of the XOR problem using the decision tree-based neural network. In: International Conference on Machine Learning and Cybernetics, vol. 2, pp. 1048–1052 (2003)
16. Barmann, F., Biegler-Konig, F.: On a class of efficient learning algorithms for neural networks. Neural Networks 5, 139–144 (1992)
17. Gad, E.F., Atiya, A.F., Shaheen, S., El-Dessouki, A.: A new algorithm for learning in piecewise-linear neural networks. Neural Networks 13, 485–505 (2000)
18. Wen, C., Ma, X.: A max-piecewise-linear neural network for function approximation. Neurocomputing 71(4-6), 843–852 (2008)

sEMG Signal Classification for the Motion Pattern of Intelligent Bionic Artificial Limb*

Yang Li[1], Yantao Tian[1,2,**], and Wanzhong Chen[1]

[1] School of Communication Engineering, Jilin University, Changchun, 130025
[2] Key Laboratory of Bionic Engineering, Ministry of Education Jilin University
tianyt@jlu.edu.cn

Abstract. Surface EMG (sEMG) signal classification based on the motion pattern plays an important role in control system design of intelligent bionic artificial limb. The key problems and the corresponding solutions of sEMG signal classification, which consist of recognition rate, algorithm complexity, robustness and real-time characteristic, were summarized in this paper. By comparing with the performance of practical application, the research directions for the future work are pointed out at last.

Keywords: intelligent bionic artificial limb, sEMG signal, recognition rate, robustness, real-time characteristic.

1 Introduction

As a new research interest, sEMG classification has attracted more and more attention in the past decades years. sEMG signal is a biomedical signal that measures electrical currents generated in muscles during its contraction representing neuromuscular activities. It is a short time stationary and nonlinear signal whose statistical characterization changes when time changes [1].

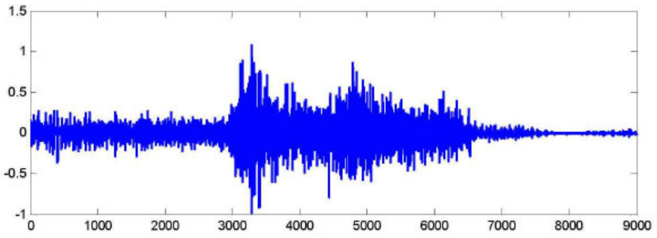


Fig. 1. Illustration of a typical sEMG signal of the upper arm. The x axis presents time, y axis presents amplitude.

---

* This paper is supported by the Key Project of Science and Technology Development Plan for Jilin Province (Grant No.20090350), Chinese College Doctor Special Scientific Research Fund (Grant No.20100061110029) and the Jilin University "985 project" Engineering Bionic Sci. & Tech. Innovation Platform.

** Corresponding author.

© Springer-Verlag Berlin Heidelberg 2011

However, sEMG signal is an exceeding complicated signal which is composed of many recruited motor units under surface electrode and background noises [2], and is easily affected by various factors. So classifying sEMG signal correctly which determines the practicability of intelligent bionic artificial limb has become the focus at home and abroad. This paper generalized the current difficulties in four points, recognition rate, algorithm complexity, robustness and real-time characteristic. By comparing the methods proposed by researchers, this paper demonstrated the actuality and possible challenges, and pointed out the research direction in the future.

2 Key Problems of sEMG Signal Classification and Solutions

Efficient classification is the key for sEMG to control intelligent bionic artificial limb. The technology that inputs features, which extracted from sEMG signal, to classifiers and controls intelligent bionic artificial limb efficiently through correct classification, is more and more mature. But it still has some difficulties in recognition rate of classifier, algorithm complexity, robustness, and real-time characteristic.

2.1 Recognition Rate

a) In 1989, A. Hiraiwa [3] used BP neural network in his paper to classify five motions by selecting 1channel sEMG, the average correct rate is 62%. Based on the low recognition rate, N. Uchida [4] improved it up to 86% by adding another electrode on the extensor digitorum.
b) Finding appropriate classifiers according to the characteristic of sEMG signal could enhance the recognition rate. Fuzzy logic systems can tolerate the contradictions in data. F. H. Y. Chan et al. [5] used fuzzy approach to classify single-site EMG signals for multifunctional intelligent bionic artificial limb control. The average recognition accuracy is above 90%. Fuzzy approach has slightly higher recognition rate, and it is insensitive to overtraining, and higher reliability which demonstrated by consistent outputs.
c) multilayer feed-forward neural network based on BP is widely used in actual project, but it has limitation in processing complicated construction and small samples which can be overcame by Elman neural network [6]. Pingao Mei et al. use Elman network to identify four motion patterns that were palmar dorsiflexion and flexion, hand opening and closing. The average experimental results can reach 92.5% [7]. From the result it shows that this method has a higher identification rate and great potential in analyzing other non-stationary physiological signal.
d) Kernel machine learning has been introduced into sEMG signal classification. Kernel machine learning can solve such as small samples, nonlinear, high dimension and the local minimum points. A novel EMG classifier called cascaded kernel learning machine (CKLM) was proposed by Yi-Hung Liu et al. [8] to achieve the goal of high-accuracy EMG recognition. SVM (shown in Table I). The best EMG recognition rate 93.54% is obtained by CKLM.

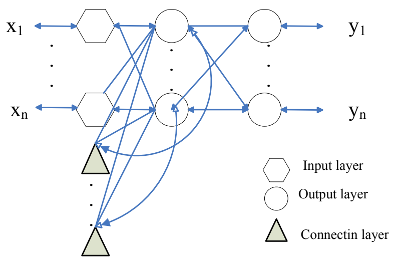


Fig. 2. A typical Elman network

Table 1. Comparisons of Average Classification Rates(%) Among Different EMG Classifiers for Three Subjects

Classifier	Subject1	Subject2	Subject3
k-NN	67.34	70.45	85.12
BP	79.21	82.78	90.43
SVM	85.13	86.78	93.74
GDA+ k-NN	84.43	86.50	92.47
GDA+ BP	86.73	87.89	93.43
GDA+ SVM(CKLM)	93.54	92.33	96.76

e) In 2008, Zhihong Liu proposed a hand motion recognition method based on EMG signals, using learning vector quantization (LVQ) neural networks[9] . Compared with BP and ART neural networks it possesses simpler network structure, faster learning rate, more reliable classification, and better fault tolerance. The recognition accuracy can reach up to 98%. Table 2 is the comparison of LVQ and Elman.

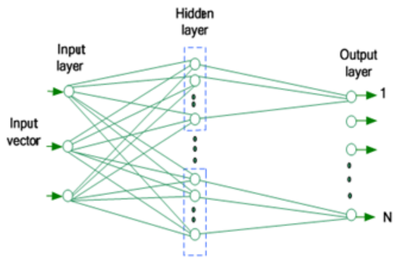


Fig. 3. The LVQ neural networks structure

Table 2. The Comparison Between LVQ and Elman Classifiers

Method	Correct Rate (%)			
	WE	WF	HE	HG
LVQ	100	100	100	96
Elman	98.3	96.6	88.3	86.6

Table 3. Illustration of different recognition rates using different classifiers

Reference	Year	Classifier	Recognition performance
A. Hiraiwa [2]	1989	BP	62%
N. Uchida [4]	1992	BP	86%
F. H. Y. Chan [14]	2000	Fuzzy approach	above 90%
Yi-Hung Liu [18]	2007	CKLM	93.54%
Pingao Mei [16]	2008	Elman	92.5%
Zhihong Liu [21]	2008	LVQ	98%

Table 3 shows the illustration of different recognition rates using different classifiers. we can see that there exist gaps between the obtained results and the requirement of practical application.

2.2 Algorithm Complexity

Recognition rate is an important criterion to judge the capability of classifiers, but the practicality of classifier is determined by algorithm complexity. In practice, a classifier which has simple algorithm and high recognition rate, has great applicability in the complicated system of intelligent bionic artificial limb controlled by sEMG signal. A simple algorithm means low complexity and fast convergence rate. By comparing LM method with variable learning rate (VLR) method (shown in Table 4), Dongjing Zhao et al. found out that although both of them have good classification for three motions, LM method with neural network is better. It has higher recognition ratio and learning efficiency and its discrimination and robustness are better [10], which means it has great potential in controlling human-computer system.

Table 4. Comparison of Learning Rate between LM and VLR

Classifier	Input Layer	Output Layer	Samples	Times	Recognition Rate
LM	8	3	18	2-3s	94.62%
VLR	8	3	18	4-5s	91.54%

Besides neural network, Pi-sigma network, which can restrict the number of weights on net, is also quickly converged and can avoid construction complicated due to the increase of input component. Haihong Zhang [11], comparing this method with BP network (shown in Table 5), found out that the learning time of Pi-sigma is much faster than BP.

Table 5. Comparison of Learning Rate between Pi-sigma and BP

Mean Square Error	0.06	0.04	0.02	0.01
BP	42s	54s	95s	159s
Pi-sigma	3	15	57	105s

High recognition rate is not the only criterion to judge the capability of classifiers. Now researchers do not pursuit higher correct rate blindly any more, but combine with the characteristic of the system itself. Therefore, the classifier with simple algorithm and relatively high recognition rate is popular with scholars.

2.3 Robustness

From early 1970', researchers have studied the classification of hand motions such as finger flexion-extension, wrist flexion-extension and supination-pronation by sensing the activities of upper arm muscles. Although the best recognition rates have reached 98% in the recent research by choosing different classifiers, the number of motions which used to classify is still too small, with only 4-6 motions, and the motions are quite different so that they can be easily recognized. It has a great distance with practical applications. On this point, multi-features and mixed signals were used to classify multiple motions.

a) Xiang Chen et al. [12] proposed a method. In his paper, The recognition results of 5,6,11,13 and 16 gestures are shown in Table 6.

Table 6. Recognition Results of Different Numbers of Gestures

Numbers of Gestures	5	6	11	13	16
Correct rate	92.1%	96.8%	90.2%	74%	85.3%

From this table, we can see that, generally these are ideal results, but the recognition rate of 13 gestures is lower than the others. That is because multi-finger extension gestures are more difficult to recognize than single finger extension gestures. The problem is more prominent when similar finger activities are involved in the different gestures. The classifier used in this paper is Bayesian classifier. In further studies, Bayesian classifier should be replaced, and the number of electrodes should be increased slightly to enhance the recognition rate.

b) At present, the recognition rate of multi-motion is relatively low according to sEMG. In order to enhance the practicality of the intelligent bionic artificial limb, the mixed signals, including perceptual signals and sEMG signals, should be used for classification, in addition to increase the number of motions which is used to classify. It can improve the reliability of classification. At the same time using the mixed signals makes the classifier have better ability of analysis. By adopting different ways to extract feature, Zhizeng Luo et al. [13] found out that input tactile and slip signals together to classifier was an efficient way to enhance the practicality and dexterity of intelligent bionic artificial limb. The recognition results of mixed signals are shown in Table 7.

Table 7. Correct Rate of Mixed Signals Used Different Classifiers

Reference	Classifier	Feature	Number of Motions	Number of Materials	Correct Rate
[14]	BP	AR	2	4	above 86%
[33]	Bayes	Power Spectrum Ration	4	–	100%for tactile 84%for slippage

Both multi-motion recognition method and mixed signal method have achieved good results. But there is still a great distance between the theory and application. It is the focus of the future work to find appropriate features and classify algorithm and to improve the robustness of classifier.

2.4 Real-Time Characteristic

Faster processing speed and better classification result have been obtained under the laboratory conditions recently, but most of them control intelligent bionic artificial limb movement by judging whether the sEMG signal is intense to achieve a certain threshold, which can not meet the users' requirement to control multi-motion movement with artificial limb, when the movement patterns increase. To let the user operates the hand without perceiving a time delay, the response time of a myoelectric artificial limb should be less than 300 ms, which requires the system a great real-time characteristic.

a) Zhizeng Luo et al. [14] firstly used wavelet analysis to eliminate the noise in the acquired EMG signals to the maximum extent. The recognition rate of 4 motions was 88%. Using this method, the intelligent bionic artificial limb's start time and duration of a movement can be synchronized with real hand. In this way, user won't perceive a time delay.
b) Although Pi-sigma has such advantages, it has a serious flaw that the number of nodes in hidden layer will influence the network performance significantly. The fact that choosing appropriate number of nodes in hidden layer suited to different situations needs to be further studied.

In 2007, JianGao et al. [15] proposed two improved SVM methods [16-21] which have global optimization, short training time and good generalization abilities in the paper, to enhance the discrimination rate, and shorten the process time.
c) Because of the cost of artificial limb, single chip microcomputer was widely used in intelligent bionic artificial limb systems, which is difficult to be carried out by using ANN or SVM. Therefore, binary tree was introduced to the system, which has simple configuration, small computation and real-time characteristic. In the paper written by Zhizeng Luo et al., five motion patterns of hand (dorsiflexion, flexion, hand opening, closing and no action) are identified by using the pattern classification of binary tree. The average recognition ratio is above 94%, while some results can reach 100% [34]. The system of two-freedom electric artificial hand using binary tree has fast response, reliable motions, good manipulation and adaptive ability. It could be used practically.

Most of the myoelectric hand researches are studied in laboratory using simulation software. In future work, to meet the need of hardware system, appropriate classifier should be selected to improve the correct rate and real-time characteristic, making intelligent bionic artificial limb can be used practically.

3 Conclusion and Prospection

Accurate recognition of the user's intention on the basis of the measured sEMG signals is the key problem in the realization of myoelectric control. At present, although the recognition rate, algorithm complexity, robustness and real-time performance of classifiers have achieved good results, there still exist some difficulties which need to be solved in the future:

1. Low recognition rate: Although some of the recognition results are above 90%, and the best correct rate can reach 98%, even 1% error rate can cause injury to users. So the correct rate should be steadily increased to higher than 95% in the future work.
2. Weak robustness: The number of motions should be increased, and perceptual signals should be input to classifier combining with sEMG signal, in order to enhance the correct rate of classification. It requires the classifier a better ability of analysis. Now the average recognition rate is only about 80%, which is much lower than practical requirement. Using mixed signals including more than two perceptual signals and sEMG signal as input to classifier, and making the result of multi-motion recognition above 90%, are the keys of future work.
3. Bad real-time characteristic: We hope that appropriate classifier which suit to single chip microcomputer or DSP can be found to enhance the real-time characteristic of system.

According to the recognition methods summarized above, we find that classifier design and application are very important in recognition successful. Besides that, choosing feature correctly and typically is also an important factor that influences the recognition rate. In order to extract the most representative feature, the corresponding relationship between sEMG and all kinds of gestures and movements should be analyzed. Therefore, it is the direction of future researches that establishing single-input, multiple-output model of sEMG signal which actually reflected system characteristics.

Only out-put signal can be obtained when establishing sEMG model. Therefore, blind identification is used to build model to recognize different patterns. In this paper, instrumental variable is tried to use when studying the corresponding blind identification methods.

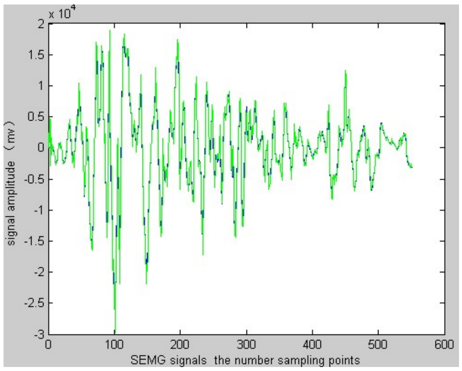


Fig. 4. Figure of Original Signal (blue) and sEMG Model (green)

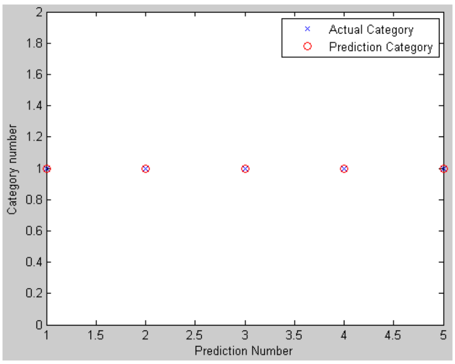


Fig. 5. Recognition Results

Figure 4 is the comparison result between model signal (the colour is blue) and original signal (the colour is green). Figure 5 is the recognition results. In this paper, only test one movement pattern. The correct rate is 100%, and the training time is only 0.20s which is faster than all the mentioned above. From these we can see that the instrumental variable with blind identification method is effective in recognizing different movement patterns.

Using instrumental variable with blind identification is the first attempt in physiological signal modeling and recognition. It has good application prospect in analyzing physiological signal because of its simpleness and less calculation. Meanwhile, the unbiased estimation of parameters can be got by using this method. It also can improve the correct recognition rate.

Using sEMG with temperature and tactile information as the controlling signal of artificial hand will improve the practicability and the capability of artificial limb. sEMG signal still mix with noise after denoising, therefore, appropriate classifier need to be found to overcome the noise interference and improve the recognition rate.

Generally, by analyzing sEMG signal we find that extracting feature effectively and recognizing correctly will arouse general interest in studying artificial limb. Selecting appropriate features and classifiers are very important in order to improve the recognition rate, robustness and real-time characteristic.

References

1. Mangieri, E., et al.: A Novel Analogue Circuit for Controlling Prosthetic Hands. In: IEEE Biomedical Circuits and Systems Conference, USA, pp. 81–84 (2008)
2. Stokes, I.A.F.: Relationships of EMG to effort in the trunk under isometric conditions force-increasing and decreasing effects and temporal delays. Clinical Biomechanics 20, 9–15 (2005)
3. Hiraiwa, A., Shimohara, K., Tokunaga, Y.: EMG Pattern Analysis and Classification by Neural Network. In: IEEE International Conference on Systems, Man and Cybernetics, Cambridge, MA, USA, pp. 1113–1115 (1989)
4. Uchida, N., Hiraiwa, A., Sonehara, N., Shimoharal, K.: EMG pattern recognition by neural networks for multi finger control. Proc. Ann. Eng. Med. Biol. Soc., 1016–1018 (1992)
5. Wang, R.C., Huang, C.H., Li, B.: A Neural Network-Based Surface EMG Motion Pattern Classifier for the Control of Prostheses. In: IEEE International Conference Engineering in Medicine and Biology Society, vol. 3, pp. 1275–1277 (1997)
6. Andrade, A.O., Soares, A.B.: EMG Pattern Recognition For Prosthesis Control. In: COBEM 2001: Brazilian Congress of Mechanical Engineering (2001)
7. Morita, S., Kondo, T., Ito, K.: Estimation of Forearm Movement from EMG Signal and Application to Prosthetic Hand. In: IEEE International Conference Robotics and Automation, vol. 4, pp. 3692–3697 (2001)
8. Wang, B., Jun, L., Bai, J., Peng, L., Li, Y., Li, G.: EEG recognition based on multiple types of information by using wavelet packet transform and neural networks. In: Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference, pp. 5377–5380 (2005)
9. Jiang, M.W., Wang, R.C., Wang, J.Z., Jin, D.W.: A Method of Recognizing Finger Motion Using Wavelet Transform of Surface EMG Signal. In: Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference, Shanghai, pp. 2672–2674 (2005)

10. Ajiboye, A.B., Weir, R.F.F.: A heuristic fuzzy logic approach to EMG pattern recognition for multifunctional prosthesis control. IEEE Transactions on Neural Systems&Rehabilitation Engineering 13, 280–291 (2005)
11. Becker, K., Thull, B., Kamacher-Leidinger, H., Stemmer, J., Rau, G., Kalff, G., Zimmermann, H.J.: Design and validation of an intelligent patient monitoring and alarm system based on a fuzzy logic process model. Artif. Intell. Med. 11, 33–53 (1997)
12. Mahfouf, M., Abbod, M.F., Linkens, D.A.: A survey of fuzzy logic monitoring and control utilisation in medicine. Artif. Intell. Med. 21, 27–42 (2001)
13. Ajiboye, A.B., Weir, R.F.F.: A Heuristic Fuzzy Logic Approach to EMG Pattern Recognition for Multifunctional Prosthesis Control. IEEE Trans. on Neural Systems and Rehabilitation Engineering 13, 280–291 (2005)
14. Chan, F.H.Y., Yang, Y.S., Lam, F.K., Zhang, Y.T., Parker, P.A.: Fuzzy EMG classification for prosthesis control. IEEE Trans. Rehabil. Eng. 8(3), 305–311 (2000)
15. Raez, M.B.I., Hussain, M.S., Mohd-Yasin, F.: Techniques of EMG signal analysis: detection, processing, classification, and applications. Biological Procedures Online, 11–35 (2006)
16. Lei, T., Wu, Z., Yang, Z.: Application of Elman Recursive neural network to structural analysis. Electric Locomotives & Mass Transit Vehicles 27, 56–58 (2004) (in Chinese)
17. Mei, P., Luo, Z.: sEMG disposal based on wavelet packet analysis and Elman neural network. Mechanical & Electrical Engineering Magazine 25, 7–10 (2008) (in Chinese)
18. Liu, Y.-H., Huang, H.-P., Weng, C.-H.: Recognition of Electromyographic Signals Using Cascaded Kernel Learning Machine. IEEE/ASME Transactions on Mechatronics 12, 253–264 (2007)
19. Liu, Z., Luo, Z.: Hand Motion Pattern Classifier Based on EMG Using Wavelet Packet Transform and LVQ Neural Networks. In: Proceedings of 2008 IEEE International Symposium on IT in Medicine and Education, pp. 28–32 (2008)
20. Ruifeng, B.: Application of learning vector quantization (LVQ) in selecting mechanism type in mechanical design. In: Proceedings of the 7th International Symposium on Test and Measurement, pp. 2692–2695 (2007)
21. Zhao, J., Jiang, L., Jin, M., Liu, H., Cai, H.: Prosthetic Hand Control Based on EMG Signal. Machinary & Electron 8, 54–56 (2006) (in Chinese)
22. Gao, J., Luo, Z.: Application of SVM in Electromyography Pattern Recognition. Chinese Journal of Sensors and Actuators 20, 366–369 (2007) (in Chinese)
23. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, N.Y (1995)
24. Cortes, C., Vapnik, V.N.: Support vector networks. Machine Learning 20, 273–297 (1995)
25. Scholkopf, B.: Support Vector Learning. Oldenbourg Verlag, Munich (1997)
26. Vapnik, V.: Statistical Learning Theory. Wiley-Interscience Publication, Hoboken (1998)
27. Smola, A.J.: Learning with Kernels. PhD thesis, Technische Universitat Berlin (1998)
28. Vapnik, V.: An overview of statistical learning theory. IEEE Transactions on Neural Networks 10, 988–999 (1999)
29. Scholkopf, B., Burges, C.J.C., Smola, A.J.: Advances in Kernel Methods—Support Vector Learning. MIT Press, Cambridge (1999)
30. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press, Cambridge (2000)
31. Campbell, C.: An introduction to kernel methods. In: Howlett, R.J., Jain, L.C. (eds.) Radial Basis Function Networks: Design and Applications, pp. 155–192. Springer, Berlin (2000)
32. Müller, K.-R., Mika, S., Ratsch, G.: An introduction to kernel-based learning algorithms. IEEE Transactions on Neural Networks 12, 181–201 (2001)
33. Scholkopf, B., Smola, A.J.: Learning with Kernels (2002)
34. Luo, Z., Li, W.: A design of two-freedom EMG artificial hand with real-time control. Chinese Journal of Rehabilitation Medicine 24, 355–358 (2009) (in Chinese)

Dynamic Construction of Multilayer Neural Networks for Classification

Jiqian Liu and Yunde Jia

Beijing Laboratory of Intelligent Information Technology,
School of Computer Science, Beijing Institute of Technology,
Beijing 100081, P.R. China
{liujiqian,jiayunde}@bit.edu.cn

Abstract. There are several drawbacks of multilayer neural networks (MLNNs) including the difficulty of determining the number of hidden nodes and their black box nature. We propose a new dynamic construction mechanism for MLNNs to overcome such inherent drawbacks. The main goal of our work is to train a hidden neuron and assemble it to the network dynamically while making the learning error smaller and smaller. In this paper, a hidden neuron carries out the function of a linear classifier which answers yes(Y) or no(N) to whether the input data belongs to the specific class. We call such a linear classifier a Y/N classifier and call the hidden neuron a Y/N neuron. The number of Y/N neurons are determined self-adaptively according to the given learning error and then successfully avoid the overlearning problem. The dynamically constructed MLNN with Y/N neurons is called a Y/N neural network. We prove that a Y/N neural network can always converge to the required solution and illustrate that Y/N neural networks can be applied to very complex classification problems.

Keywords: Multilayer neural network, Y/N neuron, Y/N classifier, Y/N neural network.

1 Introduction

Multilayer neural networks (MLNNs) have been developed for solving complicated classification problems by employing hidden layers. The additional hidden layers significantly enhance the classification capacity of the network. It has been proved that feedforward multilayer neural network with only one hidden layer can approximate any function to arbitrary accuracy if given sufficient number of hidden nodes [1]. Compared with statistical models, MLNNs can be applied without prior knowledge about the statistical distributions of classes in data sources. This makes MLNNs more suitable for some particular tasks where it is hard to make some reasonable prior assumptions. However, MLNNs have some inherent drawbacks which severely hamper their further development. One of them is how to optimize the structure of MLNNs, especially to determine the number of hidden nodes. A lot of research has been done in this area.

© Springer-Verlag Berlin Heidelberg 2011

Mirchandani and Cao [2] developed a theorem to establish a relationship between the number of hidden nodes and other neural network parameters. Geman et al. [3] discussed how the number of hidden units affects the bias/variance trade-off. Lawrence et al. [4] illustrated that minimizing the number of hidden units does not always lead to the best generalization. Bartlett [5] showed that the generalization performance depends on the size of weights rather than the number of weights. Although many methods [6] [7] [8] [9] [10] [11] have been proposed to solve the problem, in most situations, the best number of hidden nodes is still selected empirically.

In fact, when a MLNN is applied for forecasting or recognition, the distribution functions are usually unknown and samples from the distributions could come in any way. It is a really risky work to train a MLNN to approximate an unknown function by some random samples with only one control parameter. If the number of hidden nodes is too small, the classification capability of the network may be too poor to handle complex cases. If the number of hidden nodes is too large, the generalization ability of the network will be severely impaired due to the network memorizing too much detail of the class patterns in the training dataset. Besides,the black box nature, the other drawback of MLNNs, increases the difficulty of classification. When a MLNN is being trained, no useful information can be extracted out to guide the network to learn. In conclusion, it is hard to make some progress in solving the problem under the current working mechanism of MLNNs.

In this paper, we propose a novel method for constructing MLNNs dynamically to overcome these inherent drawbacks. We also present a new kind of computing neurons to substitute hidden nodes in the hidden layer of MLNNs. The new kind of neurons carries out the function of a linear classifier which answers yes(Y) or no(N) to whether the input data belongs to the specific class. We call such a linear classifier a Y/N classifier and call the neuron a Y/N neuron. The trained Y/N neurons are assembled to the middle layer of the MLNN one by one. We call the dynamically constructed MLNN with Y/N neurons a Y/N neural network. In the following section, we give the detailed description of what kind of computation is performed by a Y/N classifier. Then we discuss the construction of a Y/N neural network and how to train a Y/N neural network for very complex classification problems. In the last section, we summarize our conclusions and indicate future research.

2 Y/N Classifier

A linear classifier can be interpreted as a hyperplane that separates the input data space into two parts, the positive and negative sides of the hyperplane. We define a Y-classifier as a linear classifier on the positive side of which all data points do belong to the specific class and a N-classifier as a linear classifier on the positive side of which all data points do not. Data points on the negative side of the hyperplane are under-constrained. A Y-classifier and a N-classifier together are called a Y/N classifier. Fig. 1 gives an illustration of how a Y/N classifier works. Compared with the SVM methods [12] [13], learning a Y/N classifier only needs to find support vectors from the target class which makes it more suitable for complex cases.

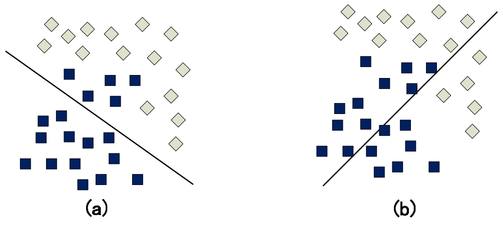


Fig. 1. Illustration of how a Y/N classifier works. The line in (a) is a Y/N classifier while the one in (b) is not.

The most essential problem here is whether there always exists a Y/N classifier for the given input dataset. We prove that the answer is yes. And we only give the proof for the two-dimensional case because it can be easily extended to the higher dimensional spaces. Fig. 2 illustrates why there always exists a Y/N classifier. Samples from class A are shown with black squares and samples not from class A are shown with gray diamonds. For finite datasets, there always is a circle, as circle C shown in Fig. 2, that passes through some points and contains all the other points in its interior. Assume circle C passes through point $\boldsymbol{P}$, we can find a circle E with its centre at point $\boldsymbol{P}$ and contains no other points inside it. Circle C and circle E intersect at two points, $\boldsymbol{M}$ and $\boldsymbol{N}$. Draw a line l_1 passing through $\boldsymbol{M}$ and $\boldsymbol{N}$. Then l_1 is a Y/N classifier.

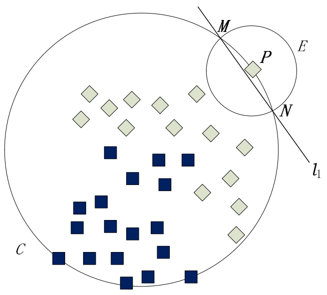


Fig. 2. Illustration of why there always exists a Y/N classifier

The next problem we are concerned with is how to design a Y/N classifier. We consider the Y-classifier case where our goal is to find a hyperplane on the positive side of which all data points belong to the specific class A. So data points not from class A are all on the negative side of this hyperplane. There are infinite hyperplanes satisfying this criterion. The best Y-classifier should be the one that can separate most of the data points in A from those not in A. This optimization problem can be expressed as

$$\max \sum_i \frac{sgn(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) + 1}{2}, \quad \boldsymbol{x}_i \in A \tag{1}$$

$$s.t. \qquad \boldsymbol{w} \cdot \boldsymbol{x}_j + b < 0, \quad \boldsymbol{x}_j \in \bar{A} \tag{2}$$

where $\boldsymbol{w}$ is the normal vector of the classification hyperplane, b the bias value and sgn the signum function. Since $sgn(x)$ is not differentiable, the optimization function (1) can not be optimized by Lagrange multiplier method. Hence we introduce the following optimization problem

$$\max \sum_i (\boldsymbol{w} \cdot \boldsymbol{x}_i + b), \quad \boldsymbol{x}_i \in A \tag{3}$$

$$s.t. \qquad -\boldsymbol{w} \cdot \boldsymbol{x}_j - b > 0, \quad \boldsymbol{x}_j \in \bar{A}.$$

The main idea is that the more points on the positive side of the classification hyperplane the larger the sum in function (3) is. We do not solve this problem directly for the reason that the solution procedure is not very suitable for neural network implementation. In the next part, we will further simplify the problem to a more tractable one.

3 Construction of a Y/N Neural Network

The basic idea of our Y/N neural network approach for classification is to find new Y/N classifiers dynamically to make the learning error smaller and smaller, as shown in Fig. 3. First, the classifier l_1 is learned from all data points. Then classifier l_2 is learned from data points on the negative side of l_1. This procedure goes on and more and more data points are classified. At last, the remain data points can be classified correctly by l_3. This procedure is similar to that of the boosting methods [14] [15] [16]. But our Y/N classifiers are fundamentally different from those statistical classifiers used in the boosting methods. In order to realize the basic idea, there are two main problems need to be addressed. The first one is what kind of neuron can work as a Y/N classifier. We call a neuron that can carry out the function of a Y-classifier(N-classifier) a Y-neuron(N-neuron). Y-neurons and N-neurons together are called Y/N neurons. The other one is how these Y/N neurons are organized in the neural network. We discuss these two problems below.

The optimization function (3) can be rewritten as

$$\max(\boldsymbol{w} \cdot \sum_i \boldsymbol{x}_i + Nb)$$

$$or \quad \max(\boldsymbol{w} \cdot E\boldsymbol{x}_i + b) \tag{4}$$

where N is the number of $\boldsymbol{x}_i$ and $E\boldsymbol{x}_i$ the mean. Function (4) suggests that the best Y/N classifier is the hyperplane on which $E\boldsymbol{x}_i$ has the maximum projection

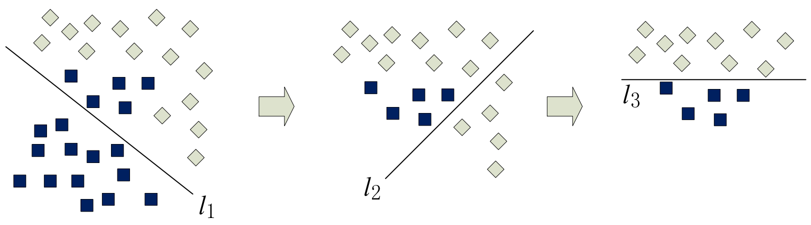


Fig. 3. Illustration of the data points being classified by a Y/N neural network gradually

value if given all data points from $\overline{A}$ are on its negative side. To simplify the resolution of this problem, we choose the mean of $\boldsymbol{x}_j$, $E\boldsymbol{x}_j$, as the reference point. Then $\boldsymbol{w}$ can be calculated by

$$\boldsymbol{w} = E\boldsymbol{x}_i - E\boldsymbol{x}_j \tag{5}$$

while b is fixed by Equation (2). Here the computations of $\boldsymbol{w}$ and b are only an approximate solution for optimization problem (4), but they are very simple and can be easily implemented by neurons.

Once a Y/N neuron is trained, it is assembled to the middle layer of the neural network. Y-neurons are linked to the output neuron with excitatory connections and N-neurons with inhibitory connections. The weight of the connection between the newly trained Y/N neuron and the output neuron is recommended to be set to one third or smaller of that of the previously trained Y/N neuron. Such a mechanism guarantees that the earliest trained Y/N neuron which is activated determines the final status of the output neuron. If the integration of Y/N neurons in the middle layer is larger than zero, the output neuron will be activated which means the input data comes from the specific class. Otherwise, the output neuron shows no response.

4 Learning in a Y/N Neural Network

Considering how to define the concept of human beings, one may claim that human beings have two legs as compared to quadrupedal mammals and have no wings as compared to birds. In this way, one can give a more and more detailed description of the concept. But one will find that it is impossible to give the exact definition of human beings. On the contrary, it is much easier to determine what is not a human being, such as a thing having four legs is not a human being or a thing having wings is not a human being. So instead of defining a very complex concept B directly, we can define it by its opposite concept $\overline{B}$. That is, if we have any opportunity to determine an object belongings to the opposite concept $\overline{B}$, the object is definitely not an object called B. Otherwise, we say that it is an object belonging to the concept of B.

The learning process described above can be implemented by a Y/N neural network when the training data are selected carefully. Fig. 4 gives an illustration

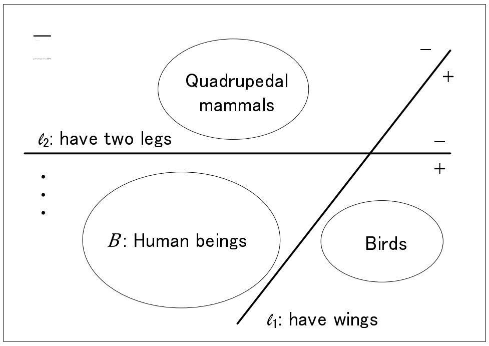


Fig. 4. Illustration of how to learn the concept of human being

of such a learning process. It should be noticed that the specific class A here is not B but $\overline{B}$, which means the output neuron will be activated only when the input pattern belongs to $\overline{B}$. As shown in Fig. 4, patterns from the class 'human beings' and the class 'birds' are first used to train the classifier l_1. Then patterns from the class 'human beings' and the class 'quadrupedal mammals' are used to train the classifier l_2. In this way, more and more classifiers are trained. By combining these trained classifiers, we get a Y/N neuron network that can respond to the concept of $\overline{B}$.

5 Conclusions

In this paper we propose the Y/N neural network model to overcome the inherent drawbacks of the traditional MLNNs. The learning error of our model becomes smaller and smaller while assembling the newly trained Y/N neurons to the network dynamically. Compared with traditional MLNNs, Y/N neural networks provide much more useful information by using Y/N neurons instead of hidden nodes. We illustrate that Y/N neural networks can be applied to very complex classification problems. Future work can be focused on the design of new kinds of Y/N classifiers and the applications of Y/N neural networks.

Acknowledgments. This work was partially supported by the Natural Science Foundation of China(NSFC) under Grant No. 90920009 and NSFC-Guangdong Joint Fund under Grant No. U1035004.

References

1. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
2. Mirchandani, G., Cao, W.: On hidden nodes for neural nets. IEEE Transactions on Circuits and Systems 36, 661–664 (1989)

3. Geman, S., Bienenstock, E., Doursat, R.: Neural Networks and the Bias/Variance Dilemma. Neural Computation 4, 1–58 (1992)
4. Lawrence, S., Giles, C.L., Tsoi, A.C.: What Size Neural Network Gives Optimal Generalization? In: Convergence Properties of Backpropagation. Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (1996)
5. Bartlett, P.L.: The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. IEEE Transaction on Information Theory 44, 525–536 (1998)
6. Ash, T.: Dynamic Node Creation in Backpropagation Networks. Connection Science 1, 365–375 (1989)
7. Elisseeff, A., PaugamMoisy, H.: Size of multilayer networks for exact learning: analytic approach. In: Michael, I., Jordan, M.C.M., Petsche, T. (eds.) Advances in Neural Information Processing Systems, vol. 9, pp. 162–168. The MIT Press, Cambridge (1997)
8. Fletcher, L., Katkovnik, S.F., Engelbrecht, A.: Optimizing the number of hidden nodes of a feedforward artificial neural network. In: IEEE International Joint Conference on Neural Networks, vol. 2, pp. 1608–1612 (1998)
9. Sarle, W.S.: Stopped training and other remedies for overfitting. In: 27th Symposium on the Interface, pp. 352–360 (1995)
10. Xu, L.: Bayesian Ying-Yang System and Theory as A Unified Statistical Learning Approach (III) Models and Algorithms for Dependence Reduction, Data Dimension Reduction, ICA and Supervised Learning. In: Wong, K.M., King, I., Yeung, D.-Y. (eds.) Theoretical Aspects of Neural Computation: A Multidisciplinary Perspective, pp. 43–60. Springer, Heidelberg (1997)
11. Xu, S., Chen, L.: A novel approach for determining the optimal number of hidden layer neurons for FNN's and its application in data mining. In: 5th International Conference on Information Technology and Applications, pp. 683–686 (2008)
12. Vapnik, V.: Support-vector networks. Machine Learning 20, 273–297 (1995)
13. Platt, J.: Sequential minimal optimization: A fast algorithm for training support vector machines. Advances in Kernel MethodsSupport Vector Learning 208, 1–21 (1998)
14. Valiant, L.G.: A theory of the learnable. Communications of the ACM 27, 1134–1142 (1984)
15. Schapire, R.E.: The Strength of Weak Learnability. In: 30th Annual Symposium on Foundations of Computer Science, pp. 28–33 (1989)
16. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: a new explanation for the effectiveness of voting methods. The Annals of Statistics 26, 1651–1686 (1998)

Existence of Periodic Solutions for Cohen-Grossberg Neural Networks with Time-Varying Delays and Impulses*

Chunxue Wu

School of Mathematics and Information, Yantai University
Yantai, Shandong 264005, P.R. China
wuchunxue1020@163.com

Abstract. By using the continuation theorem of Mawhin's coincidence degree theory, some new sufficient conditions are obtained for the existence of periodic solution of higher-order Cohen-Grossberg type neural networks with variable delays and impulses, moreover, the monotonicity and smoothness of activation function are not assumed in this paper.

Keywords: Cohen-Grossberg neural networks; periodic solutions; impulse; coincidence degree; delays.

1 Introduction

Over the past few years, the Cohen-Grossberg neural network has been widely studied. the Cohen-Grossberg neural network was introduced by Cohen and Grossberg[1], They have found important applications in signal precessing, especially in staticimage treatment, Processing of moving images requires the introduction of delay in the signals transmitted among thecells[2]. In recent years, Cohen-Grossberg neural networks with time delays has widey studied [3]-[7]. Bai [3] investigate the existence of periodic solution of Cohen-Grossberg type neural networks model with impulses by using continuation theorem of coincidence degree theory and some new analysis techniques.

In this paper, we will study the existence of periodic solution of the following Cohen-Grossberg neural networks with time-varying delays and impulses

$$\begin{cases} x_i'(t) = -d_i(x_i(t))\Big[a_i(t)x_i(t) - \sum\limits_{j=1}^{n} b_{ij}(t)f_j\left(x_j(t)\right) - \sum\limits_{j=1}^{n} c_{ij}(t)f_j(x_j\left(t-\tau_j(t)\right)) \\ -\sum\limits_{j=1}^{n}\sum\limits_{l=1}^{n} b_{ijl}(t)f_j(x_j\left(t-\tau_j(t)\right))f_l(x_l\left(t-\tau_l(t)+I_i(t)\right))\Big], t>0,\ t\neq t_k, \\ \triangle x_i(t_k) = x_i(t_k^+) - x_i(t_k^-) = -\gamma_{ik}(x_i(t_k)),\ i=1,\cdots,n,\ k\in\mathbb{Z}^+. \end{cases} \tag{1}$$

* This work is supported by the National Science Foundation of China under grant 10801113, Shandong Province Natural Science Foundation under Grant number R2010AM012.

© Springer-Verlag Berlin Heidelberg 2011

where $x_i(t)$ are the state of the ith neurons at the time t; a_{ij}, b_{ij} and b_{ijl} are the first and second-order connection weights of the neural networks, respectively; time delays τ_j are nonnegative constants; f_j and f_l are activation functions; we always assume that$a_i(t)$,$a_{ij}(t)$, $b_{ij}(t)$ $b_{ijl}(t)$,$I_i(t)$are continuous ω-periodic functions. $\triangle x_i\left(t_k\right)$ are the impulses at moments t_k and $t_1 < t_2 < \cdots$ is a strictly increasing sequence such that $lim_{k\to\infty} t_k = +\infty$, $i = 1, \cdots, n$.

For $\tau = \max\limits_{1\le j\le n, t\in[0,\omega]} \tau_j(t)$, $C([-\tau, 0], R^n)$ denotes the family of continuous functions $\varphi = (\varphi_1, \varphi_2, \cdots, \varphi_n)^T$ from $[-\tau, 0]$ to R^n with the norm

$$\|\varphi\| = \max_{1\le i\le n} \sup_{-\tau\le s\le 0} |\varphi_i(s)|.$$

As usual in the theory of impulsive differential equations, at the points of discontinuity t_k of the solution $t \mapsto x_i(t)$, we assume that $x_i(t_k) \equiv x_i(t_k^-)$, It is clear that, in general, the derivatives $x_i'\left(t_k\right)$ do not exist. On the other hand, according to the first equality in (1) there exist the limits $x_i'\left(t_k^{\mp}\right)$. According to the above convention, we assume $x_i'\left(t_k\right) \equiv x_i'\left(t_k^-\right)$.

Throughout this paper, we assume that

(H$_1$) There exists number $G_i, L_i > 0$ such that

$$f_i(x) < G_i, |f_i(x) - f_i(y)| \le L_i|x - y|, \forall x, y \in R, i = 1, 2, \cdots, n.$$

(H$_2$) $d_i(x)$ is positive and bounded, $0 < \underline{d}_i \le d_i(x) \le \overline{d}_i, i = 1, 2, \cdots, n;$

(H$_3$) There exists positive integer q such that $t_{k+q} = t_k + \omega, \gamma_{i(k+q)}(x) = \gamma_{ik}(x)$, $k = 1, 2, \cdots$.

(H$_4$) $a_i(t)$, $b_{ij}(t)$, $c_{ij}(t)$ and $\tau_j(t)$,$I_i(t)$ are all continuous periodic functions with the period ω, and the delays $0 \le \tau_j \le \tau, (, j = 1, \cdots, n)$ are bounded, and $0 \le \tau_{ji}'(t) < 1$.

The organization of this paper is as follows. In Section 2, we introduce some lemma needed in later sections. In Section 3, we prove the existence of the periodic solutions. Moreover, the monotonicity and smoothness of activation functions are not assumed in this paper, nor is the symmetric connection requirement.

2 Preliminaries

In this section, based on the Mawhin's continuation theorem, we shall study the existence of at least one periodic solution of (1). To do so, we shall make some preparations.

Let X and Z be two Banach space. Suppose that linear mapping $L : \mathrm{Dom}L \subset X \to Z$ is a Fredholm operator with index 0, and $N : X \to Z$ is continuous, and there exist both continuous projects operators $P : X \to \ker L$ and $Q : Z \to Z/\mathrm{Im}L$ such that $X = \ker P \oplus \ker L$, $Z = \mathrm{Im}Q \oplus \mathrm{Im}L$, which satisfy $\mathrm{Im}P = \ker L$ and $\mathrm{Im}L = \ker Q = \mathrm{Im}(I - Q)$. If we define $L_P : \mathrm{Dom} \cap \ker P \to \mathrm{Im}L$ by $L_P = L|_{\mathrm{Dom}L\cap\ker P} : (I - P)X \to \mathrm{Im}L$, then it is obviously that L_P is one to one and reversible. Write inverse mapping K_P, and denote $K_{P,Q} : Z \to \mathrm{Dom}L\cap\ker P$

is the generalized inverse of L_P by $K_{P,Q} = K_P(I-Q)$. Let Ω be a bounded open set in X. If $QN(\overline{\Omega})$ is bounded and $K_P(I-Q)(\overline{\Omega})$ is compact on $\overline{\Omega}$, then $N : \overline{\Omega} \to X$ is said to be $L-$compact.

Lemma 1. *(Gaines-Mawhin continuation theorem, Gaines and Mawhin [8]) Let X and Z be two Banach space, $L : \mathrm{Dom}L \subset X \to Z$ a Fredholm operator with index 0, Ω a bounded open set in X, $N : \overline{\Omega} \to Z$ $L-$compact in $\overline{\Omega}$. Suppose that*
(1) $Lx \neq \lambda Nx$ for all $x \in \partial\Omega \cap \mathrm{Dom}L$ and $\lambda \in (0,1)$;
(2) $QNx \neq 0$ for all $x \in \partial\Omega \cap \ker L$;
(3) $\deg\{QN, \Omega \cap \ker L, 0\} \neq 0$.
Then equation $Lx = Nx$ has at lest one solution in $\mathrm{Dom}L \cap \overline{\Omega}$.

For any nonnegative integer q, let

$$C\left[0,\omega;t_1,\cdots,t_q\right] = \left\{ u:[0,\omega]\to\mathbb{R}^n \mid \begin{array}{l} u(t) \quad \text{is continuous with respect to} \quad t\neq t_1,\cdots,t_q; \\ u(t+0) \quad \text{and} \quad u(t-0) \quad \text{exist at} \quad t_1,\cdots,t_q; \\ u\left(t_k\right) = u\left(t_k-0\right), \quad k=1,\cdots,q. \end{array} \right\}.$$

Let $X = \{x \in C\left[0,\omega;t_1,\cdots,t_q\right] \mid x(0)=x(\omega)\}$, $Z = X \times \mathbb{R}^{n\times(q+1)}$, and $\|z\|_Z = \|x\|_q + \|y\|$,$z=(x,y)\in Z$ with $x\in X, y\in\mathbb{R}^{n\times(q+1)}$, then it is standard to show that both X and Z are Banach spaces.

Let $r(t)$ be a ω-periodic continuous function defined on $\mathbb{R}$. We define $r^- = \min\limits_{0\le t\le\omega}|r(t)|$, $\quad r^+ = \max\limits_{0\le t\le\omega}|r(t)|$, $\quad \overline{r} = \frac{1}{\omega}\int\limits_0^\omega r(t)\mathrm{d}t$, $\quad \|r\|_2 = \left(\int\limits_0^\omega |r(t)|^2\mathrm{d}t\right)^{\frac{1}{2}}$.

3 Existence of Periodic Solution

Theorem 1. *Assume that (H_1)—(H_4) holds. Then system (1) has at least one ω-periodic solution.*

Proof. In order to use continuation theorem of coincidence degree theory to establish the existence of an $\omega-$periodic solution of (1), we take

$$L:\mathrm{Dom}L\cap X\to Z, Lx = \left(x', \Delta x\left(t_1\right),\cdots,\Delta x\left(t_q\right),0\right)$$

$$\mathrm{Dom}L = \left\{x(t)\in C\left[0,\omega;t_1,\cdots,t_q\right] \mid x(0)=x(\omega)\right\}, \quad N:X\to Z,$$

$$N(x(t)) = \left(A_i(t), \triangle x_i\left(t_1\right), \triangle x_i\left(t_2\right),\cdots,\triangle x_i\left(t_q\right)\right)$$

where

$$A_i(t) = -d_i(x_i(t))\Big[a_i(t)x_i(t) - \sum_{j=1}^n b_{ij}(t)f_j\left(x_j(t)\right) - \sum_{j=1}^n c_{ij}(t)f_j(x_j\left(t-\tau_j(t)\right))$$
$$- \sum_{j=1}^n\sum_{l=1}^n b_{ijl}(t)f_j(x_j\left(t-\tau_j(t)\right))f_l(x_l\left(t-\tau_l(t)+I_i(t)\right))\Big] \quad i=1,\cdots,n,$$

It is not difficult to show that

$$\ker L = \{x \in X \mid x = h \in \mathbb{R}^n\},$$

$$\mathrm{Im}L = \left\{ z = (f, C_1, C_2, \cdots, C_q, d) \in \mathbb{Z} \mid \frac{1}{\omega} \int\limits_0^\omega f(s)\mathrm{d}s + \sum_{k=1}^q C_k + d = 0 \right\}.$$

and ImL is closed in Z. Therefore, L is a Fredholm mapping of index zero. Take

$$P : X \to \ker L, \quad Px = \frac{1}{\omega} \int\limits_0^\omega x(t)\mathrm{d}t;$$

$$Q : X \to Z, \quad Qz = \left\{ \frac{1}{\omega} \left[\int\limits_0^\omega f(s)\mathrm{d}s + \sum_{k=1}^q C_k + d \right], 0, \cdots, 0, 0 \right\}.$$

It is trivial to show that P and Q are continuous project operators.

Since $\dim\ker L = n = \mathrm{codimIm}L$. Therefore, L is a Fredholm operator with the index 0. Hence, Im$P = \ker L$, Im$L = \ker Q = \mathrm{Im}(I - Q)$. the generalized inverse $K_{P,Q}$ exists. Furthermore, we have that N is L-compact on $\overline{\Omega}$, see Gaines and Mawhin [8].

Now it needs to show that there exists an domain Ω, which satisfies all the requirements given in corresponding to operator equation $Lx = \lambda Nx$, $\lambda \in (0, 1)$. We have

$$\begin{cases} x_i'(t) = \lambda A_i(t), t > 0, \ t \neq t_k, \\ \triangle x_i\left(t_k\right) = -\lambda \gamma_{ik}\left(x_i\left(t_k\right)\right), \ i = 1, \cdots, n, \ k \in \mathbb{Z}^+. \end{cases} \tag{2}$$

Suppose that $x(t) = (x_1(t) \cdots, x_n(t),)^T \in X$ is a solution of system (2) for a certain $\lambda \in (0, 1)$. Integrating (2) over the interval $[0, \omega]$, we obtain

$$\int\limits_0^\omega A_i(t)\mathrm{d}t - \sum_{k=1}^q \gamma_{ik}\left(x_i\left(t_k\right)\right) = 0,$$

Hence Multiplying both sides of system (2) by $x_i(t)$ and integrating over $[0, \omega]$, since

$$\begin{aligned} \int\limits_0^\omega x_i(t)x_i'(t)\mathrm{d}t &= \tfrac{1}{2} \sum_{\ell=1}^q \left[x_i^2\left(t_\ell\right) - x_i^2\left(t_\ell^+\right)\right] \\ &= \tfrac{\lambda}{2} \sum_{k=1}^q \gamma_{ik}(2 - \lambda\gamma_{ik}) x_i^2\left(t_k\right). \end{aligned}$$

we obtain

$$\begin{aligned}\underline{d}_i a_i^- \int_0^\omega |x_i(t)|^2\,\mathrm{d}t &= \int_0^\omega d_i(x_i(t))a_i(t)x_i^2(t)\mathrm{d}t + \tfrac{\lambda}{2}\sum_{k=1}^q \gamma_{ik}(2-\lambda\gamma_{ik})x_i^2\,(t_k)\\ &= \int_0^\omega d_i(x_i(t))\Big[a_i(t)x_i(t) - \sum_{j=1}^n b_{ij}(t)f_j\,(x_j(t))\\ &\quad - \sum_{j=1}^n c_{ij}(t)f_j(x_j\,(t-\tau_j(t)))\\ &\quad - \sum_{j=1}^n\sum_{l=1}^n b_{ijl}(t)f_j(x_j\,(t-\tau_j(t)))f_l(x_l\,(t-\tau_l(t)+I_i(t)))\Big]\mathrm{d}t\\ &\le \overline{d}_i \int_0^\omega \sum_{j=1}^n L_j\left(b_{ij}^+||x_j(t)+f_j(0)| + c_{ij}^+||x_j(t-\tau_j(t))|+f_j(0)|\right)\\ &\quad |x_i(t)|\mathrm{d}t + \overline{d}_i\int_0^\omega \sum_{j=1}^n\sum_{l=1}^n b_{ijl}^+G_l|L_j|x_j(t-\tau_j(t))|\\ &\quad + f_j(0)||x_i(t)| + |I_i(t)||x_i(t)|\mathrm{d}t\end{aligned}$$

On the other hand, we have

$$\int_0^\omega |x_j(t-\tau_j(t))|^2\mathrm{d}t \le k_j^2\int_0^\omega |x_j(t)|^2\mathrm{d}t$$

Then we have

$$\begin{aligned}\underline{d}_i a_i^- \int_0^\omega |x_i(t)|^2\,\mathrm{d}t \le \overline{d}_i\Big[&\sum_{j=1}^n L_j\left(b_{ij}^+ + c_{ij}^+k_j + \sum_{l=1}^n b_{ijl}^+G_lk_j\right)\|x_i\|_2\|x_j\|_2\\ &+\left(\sum_{j=1}^n |f_j(0)|\sqrt{\omega}\left(b_{ij}^+ + c_{ij}^+ + \sum_{l=1}^n b_{ijl}^+G_l\right) + \overline{I}_i\sqrt{\omega}\right)\|x_i\|_2\Big]\end{aligned}$$

write

$$\Gamma_i = \underline{d}_i a_i^- - \overline{d}_i\sum_{i=1}^n L_i\left(b_{ji}^+ + c_{ji}^+k_i + \sum_{l=1}^n b_{jil}^+k_iG_l\right)$$

$$\Delta_i = \overline{d}_i\left(\sum_{i=1}^n |f_j(0)|\sqrt{\omega}\left(b_{ij}^+ + c_{ij}^+\sum_{l=1}^n b_{ijl}^+G_l\right) + \overline{I}_i\sqrt{\omega}\right)$$

Let $A = \max\limits_{1\le i\le n}\Gamma_i$,$B = \min\limits_{1\le i\le n}\Delta_i$. we have

$$\|x_i\|_2 \le -\frac{B}{A} \doteq M_i.$$

Let $\xi_i \in [0,\omega](\neq t_k)$, $k = 1,\cdots,q$ such that $x_i(\xi_i) = \inf\limits_{t\in[0,\omega]} x_i(t)$, $i = 1,\cdots,n$. Then, by Hölder inequality, we have

$$\begin{aligned}
&x_i(\xi_i)\left[\int_0^\omega d_i(x_i(s))a_i(s)\mathrm{d}s + \sum_{k=1}^q \gamma_{ik}\right]\\
&\le \int_0^\omega d_i(x_i(s))a_i(s)x_i(s)\mathrm{d}s + \sum_{k=1}^q \gamma_{ik}x_i(t_k)\\
&= \int_0^\omega d_i(x_i(s))\Big[\sum_{j=1}^n b_{ij}(t)f_j\left(x_j(t)\right) - \sum_{j=1}^n c_{ij}(t)f_j(x_j\left(t-\tau_j(t)\right))\\
&\quad - \sum_{j=1}^n\sum_{l=1}^n b_{ijl}(t)f_j(x_j\left(t-\tau_j(t)\right))f_l(x_l\left(t-\tau_l(t)+I_i(t)\right))\Big]\mathrm{d}s\\
&\le \overline{d}_i\Big[\sum_{j=1}^n b_{ij}^+L_j\int_0^\omega |x_i(s)|\mathrm{d}s + \sum_{j=1}^n c_{ij}^+L_j\int_0^\omega |x_j(s-\tau_j)|\mathrm{d}s\\
&\quad +\omega\sum_{j=1}^n (b_{ij}^+ + c_{ij}^| f_j(0)| + \sum_{j=1}^n\sum_{l=1}^n b_{ijl}^+G_lL_j\int_0^\omega |x_j(s-\tau_j)|\mathrm{d}s\\
&\quad +\omega\sum_{j=1}^n\sum_{l=1}^n b_{ijl}^+G_l|f_j(0)| + \int_0^\omega |I_i(s)|\mathrm{d}s\\
&= \overline{d}_i\left[\sum_{j=1}^n A_{ij}\|x_j\|_2 + B_i\right]
\end{aligned}$$

where

$$A_{ij} = \sqrt{\omega}L_j\left(b_{ij}^+ + c_{ij}^+ + \sum_{l=1}^n b_{ijl}^+G_lk_j\right)$$

$$B_i = \omega\sum_{j=1}^n(b_{ij}^+ + c_{ij}^+ + \sum_{l=1}^n b_{ijl}^+G_l)|f_j(0)| + \sqrt{\omega}\|I_i\|_2$$

Hence

$$x_i(\xi_i) \le \frac{\overline{d}_i}{\underline{d}_i\omega\overline{a}_i + \sum\limits_{k=1}^q \gamma_{ik}}\left(\sum_{j=1}^n A_{ij}\|x_j\|_2 + B_i\right) \doteq D_i$$

Similarly, let $\eta_i \in [0,\omega](\neq t_k)$, $k = 1,\cdots,q$, such that $x_i(\eta_i) = \inf\limits_{t\in[0,\omega]} x_i(t)$, $i = 1,\cdots,n$.

$$x_i(\eta_i) \ge -\frac{\overline{d}_i}{\underline{d}_i\omega\overline{a}_i + \sum\limits_{k=1}^q \gamma_{ik}}\left(\sum_{j=1}^n A_{ij}\|x_j\|_2 + B_i\right) \doteq -D_i$$

Set $t_0 = t_0^+ = 0$, $t_{q+1} = \omega$. From (2), we have

$$\begin{aligned}\int_0^\omega |x_i'(t)|\,\mathrm{d}t &= \sum_{k=1}^{q+1} \int_{t_{k-1}+0}^{t_k} |x_i'(t)|\,\mathrm{d}t + \sum_{k=1}^{q} \left| x_i\left(t_k^+\right) - x_i\left(t_k\right)\right| \\ &\le \overline{d}_i a_i^+ \int_0^\omega |x_i(t)|\,\mathrm{d}t + \overline{d}_i \sum_{j=1}^n \int_0^\omega b_{ij}^+ \left(L_j|x_j(t)| + |f_i(0)\right)\mathrm{d}t \\ &\quad + \overline{d}_i \sum_{j=1}^n \int_0^\omega c_{ij}^+ \left(L_j|x_j(t-\tau_j(t))| + |f_i(0)\right)\mathrm{d}t + \overline{d}_i \omega I_i^+ \\ &\quad + \overline{d}_i \sum_{j=1}^n \sum_{l=1}^n \int_0^\omega b_{ijl}^+ G_l \left(L_j|x_j(t-\tau_j(t))| + |f_i(0)\right)\mathrm{d}t + \sum_{k=1}^q \overline{\gamma}_{ik} \\ &\doteq S_i.\end{aligned} \tag{3}$$

For $t \in [0, \omega]$,

$$|x_i(t)| \le |x_i(\xi_i)| + \int_0^\omega |x_i'(s)|\mathrm{d}s < D_i + S_i.$$

Now we take $\Omega = \left\{x = (x_1, x_2, \cdots, x_n)^T \in X | \|x\| \le Z_i\right\}$. $Z_i = D_i + S_i$, It is clear that Ω verifies the requirement (1) in Lemma 1. When $x \in \partial\Omega \cap \mathbb{R}^n$, x is a constant vector in $\mathbb{R}^n$ with $\|x\| = \sum_{i=1}^n |x_i| = Z$, where $Z > 0$ is taken sufficiently large so that

$$\begin{aligned}\min_{1\le i\le n} &\left(\underline{d}_i a_i^- - \tfrac{1}{2}\sum_{j=1}^n (b_{ij}^+ + c_{ij}^+)\underline{d}_i L_j + \tfrac{1}{\omega}\sum_{k=1}^q \gamma_{ik} - \tfrac{1}{2}\underline{d}_i \sum_{j=1}^n\sum_{l=1}^n G_l b_{ijl}^+ L_j\right) Z \\ &- \max_{1\le i\le n} \underline{d}_i \left(\sum_{j=1}^n (b_{ij}^+ + c_{ij}^+ + \sum_{l=1}^n G_l b_{ijl}^+)|f_i(0)| + I_i^+\right) > 0\end{aligned}$$

Then

$$\begin{aligned}(QNx)_i \quad &= \Big(- d_i(x_i)\tfrac{1}{\omega}\Big[x_i \int_0^\omega a_i(s)\mathrm{d}s - \sum_{j=1}^n \int_0^\omega f_j(x_j)(b_{ij}(s) + c_{ij}(s)) \\ &+ \int_0^\omega I_j(s)\mathrm{d}s - \sum_{j=1}^n\sum_{l=1}^n f_j(x_j) f_l(x_l)\tfrac{1}{\omega}\overline{b}_{ijl}\Big] - \tfrac{1}{\omega}\sum_{k=1}^q \gamma_{ij}x_i, 0, \cdots, 0, 0\Big)\end{aligned}$$

Therefore

$$\begin{aligned}x^T(QNx) &= -\sum_{i=1}^n \Big(- d_i(x_i)\tfrac{1}{\omega}\Big[x_i^2 \int_0^\omega a_i(s)\mathrm{d}s - x_i \sum_{i=1}^n \int_0^\omega f_j(x_j)(b_{ij}(s) + c_{ij}(s)) \\ &\quad + x_i \int_0^\omega I_j(s)\mathrm{d}s - x_i \sum_{i=1}^n\sum_{l=1}^n f_j(x_j) f_l(x_l)\tfrac{1}{\omega}\overline{b}_{ijl}\Big] + \tfrac{1}{\omega}\sum_{k=1}^q \gamma_{ij}x_i^2\Big) \\ &\le \sum_{i=1}^n \Bigg[\underline{d}_i a_i^- x_i^2 - \tfrac{1}{2}\sum_{j=1}^n (b_{ij}^+ + c_{ij}^+)\underline{d}_i L_j (x_i^2 + x_j^2) + \tfrac{1}{\omega}\sum_{k=1}^q \gamma_{ik} x_i^2 \\ &\quad - \tfrac{1}{2}\underline{d}_i \sum_{j=1}^n\sum_{l=1}^n G_l b_{ijl}^+ L_j (x_i^2 + x_j^2)\Bigg]\end{aligned}$$

$$
\begin{aligned}
&+\sum_{l=1}^{n}\underline{d}_i\left[\sum_{j=1}^{n}(b_{ij}^{+}+c_{ij}^{+}+\sum_{l=1}^{n}G_l b_{ijl}^{+})|f_i(0)|+I_i^{+}\right]|x_i| \\
\leq& -\sum_{i=1}^{n}|x_i|\left[\min_{1\leq i\leq n}\left(\underline{d}_i a_i^{-}-\frac{1}{2}\sum_{j=1}^{n}(b_{ij}^{+}+c_{ij}^{+})\underline{d}_i L_j+\frac{1}{\omega}\sum_{k=1}^{q}\gamma_{ik}\right.\right.\\
&\left.-\frac{1}{2}\underline{d}_i\sum_{j=1}^{n}\sum_{l=1}^{n}G_l b_{ijl}^{+}L_j\right)|x_i| \\
&\left.-\max_{1\leq i\leq n}\underline{d}_i\left(\sum_{j=1}^{n}(b_{ij}^{+}+c_{ij}^{+}+\sum_{l=1}^{n}G_l b_{ijl}^{+})|f_i(0)|+I_i^{+}\right)\right] \\
<& \; 0.
\end{aligned}
$$

Define a continuous functions $H : \mathrm{Dom}L \times [0,1] \to X$ by $Hx = -\mu x + (1-\mu)QNx$, where $u \in \partial\Omega \cap \mathbb{R}^n$ is a constant vector in $\mathbb{R}^n$ and $\mu \in [0,1]$. Thus, $\|H(x_1,\cdots,x_n,\mu)\| > 0$. As a result, we have $\deg\{QN, \Omega \cap \ker L, 0\} = \deg\{-x, \Omega \cap \ker L, 0\} \neq 0$. Condition (3) of Lemma 1 is also satisfied.

We now know that Ω satisfies all the requirements in Lemma 1. Therefore, equation (2) has at least a continuous ω periodic solutions.

This completes the proof of the theorem. □

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), 815–821 (1983)
2. Bainov, D.D., Simeonov, P.S.: Stability Theory of Differential Equations with Impulse Effects: Theory and Applications. Ellis Horwood, Chichester (1989)
3. Bai, C.Z.: Global exponential stability and existence of periodic solution of Cohen-Grossberg type neural networks with delays and impulses. Nonlinear Analysis, RWA (9), 747–761 (2008)
4. Li, Y.K.: Existence and stability of periodic solutions for Cohen-Grossberg neural networks with igh-order BAM neural networks with time-varying delays. Neural Networks (19), 1581–1590 (2006)
5. Liu, B.W., Huang, L.H.: Existence and exponential stability of periodic solutions for a class of Cohen-Grossberg neural networks with time-varying delays. Chaos Solitons and Fractals (32), 617–627 (2007)
6. Jiang, H.J., Cao, J.D.: BAM-type Cohen-Grossberg neural networks with time delays. Mathematical and Computer Modelling (47), 92–103 (2008)
7. Liu, J.: Stability of Cohen-Grossberg Neural Networks with time-varying delay. Chinese Journal of Engineering Mathematics 26(2), 243–250 (2009)
8. Gaines, R.E., Mawhin, J.L.: Coincidence Degree and Nonlinear Differential Equations. Springer, New York (1977)

Telecommunications Data Forecasting Based on a Dynamic Neuro-fuzzy Network

Paris A. Mastorocostas and Constantinos S. Hilas

Department of Informatics and Communications
Technological Educational Institute of Serres
Terma Magnisias, 62124, Serres, Greece
{Mast,chilas}@teiser.gr

Abstract. In this work a dynamic neuro-fuzzy network (DyNF-Net) is proposed, which is applied on the outgoing telephone traffic of a large organization. It is a modified Takagi-Sugeno-Kang fuzzy neural network, where the consequent parts of the fuzzy rules are neural networks with internal recurrence, thus introducing dynamics to the overall system. Real world telecommunications data are used in order to compare the DyNF-Net to well-established forecasting models. The comparison highlights the particular characteristics of the proposed neuro-fuzzy network.

Keywords: dynamic neuro-fuzzy network, telecommunications data, non-linear time series forecasting.

1 Introduction

Forecasting is a valuable aid for telecommunications managers and is used for network traffic management, infrastructure optimization and planning, and the scenario planning process. To successfully manage their business, carriers must rely on data to monitor, analyze and optimize their systems in order to map future trends and usage patterns. Charging and billing are vital in telecommunications business as the primary motive for telecommunications service provision is profit. However, the aim of the telecommunications managers is not only the maximization of profit but also the reduction of unnecessary cost. Making use of the historical data, they may predict the future demand by creating a reasonably accurate forecast of the call volume.

A case of such an organization is a University Campus with more than 6000 employees. Due to the continuous increase of the faculty members and staff, new telephone numbers are added daily, and an increasing demand for outgoing trunks exists. It is obvious that the changes in call volume are vital to the planning of future installations. The University holds an extended database, made by the Call Detail Records (CDR) of the Private Branch Exchange (PBX), which includes information such as the call origin, the area code and exchange, and the duration of each telephone call. The database is mainly used to determine the total number, as well as the number of the national, the international and the mobile calls per employee per month. It is noticed that the call classification, into different categories, reveals certain and different patterns between destinations. In particular, calls to mobile destinations are

© Springer-Verlag Berlin Heidelberg 2011

subject to higher tariffs and demonstrate an increasing trend during the last decade. This increasing traffic to mobile destinations may be used as a negotiation tool by the University's managers to negotiate the tariffs with national service providers. In the past, the forecasting ability of well established statistical methods on the University's call traffic have been studied [1]. Linear models are also suggested for forecasting trends in telecommunications data by the ITU Recommendation E.507 [2].

In this perspective, a Dynamic Neuro-Fuzzy Network (DyNF-Net) is proposed and its performance is compared with familiar forecasting approaches; namely a series of seasonally adjusted linear extrapolation methods, Exponential Smoothing Methods and the SARIMA method. All comparisons are performed on real world data.

The rest of the paper is organized as follows: in Section 2, a brief presentation of the classical forecasting methods that are compared with our proposed model is given. In Section 3, the Dynamic Neuro-Fuzzy Network is presented. The training algorithm used to train the model is described in Section 4. In Section 5 the data used in the paper and the outcome of the comparative analysis of the methods are presented.

2 Statistical Forecasting Methods

In this section the time series analysis methods that were used to compare and evaluate our proposed DyNF-Net are briefly presented. First a simple method to forecast future values of the time series was used. This method, which is known as Naïve Forecast 1 (NF1) [3], takes the most recent observation as a forecast for the next time interval. Another simple method which takes into account the seasonal factors was applied. First, the seasonality is removed from the original data and the remaining trend-cycle component is used to forecast the future values of the series by means of linear extrapolation. Then, the projected trend-cycle component is adjusted with the use of the identified seasonal factors [1]. When multiplicative seasonality is assumed we have the LESA-M (Linear Extrapolation with Seasonal Adjustment - Multiplicative), while in the case of additive seasonality we have the LESA-ADD.

A familiar group of time series analysis methods are the exponential smoothing methods. In exponential smoothing a particular observation of the time series is expressed as a weighted sum of the previous observations. The weights for the previous data values are terms of a geometric series and get smaller as the observations move further into the past. Simple Exponential Smoothing (SES) applies to processes without trend. In order to accommodate linear trend, C. E. Holt (1957) modified the simple exponential smoothing model. Winters (1960) extended Holt's method in order to cope with seasonal data. Multiplicative seasonal models (Winters MS) as well as additive seasonal models (Winters AS) exist [4].

Although linear trend represents an improvement on simple exponential smoothing, it cannot cope with more complex types of trend. Other modifications of SES can be applied to time series that exhibit damped trend. A damped trend refers to a regression component for the trend in the updating equation which is expressed by means of a dampening factor. As before an exponential smoothing model with damped trend and additive seasonality (DAMP AS) and its multiplicative seasonality counterpart

(DAMP MS) exists. One may also try to fit a damped trend model on time series with no seasonality (DAMP NoS). For a comprehensive review on exponential smoothing methods, readers are referred to the work of Gardner [4]. The above are popular in industry due to their simplicity and the accuracy that can be obtained with minimal effort in model identification.

Another familiar method to analyze stationary univariate time series data was developed by G. Box and G. Jenkins. The method, which is called Auto Regressive Integrated Moving Average method (ARIMA), presumes weak stationarity, equally spaced intervals or observations, and at least 30 to 50 observations. The seasonal ARIMA (SARIMA) also exists [5].

To evaluate the amount of forecast error and compare the models three statistics are used namely the root mean squared error (RMSE), the mean absolute percentage error (MAPE), and Theil's U [3].

3 Dynamic Neurofuzzy Network (DyNF-Net)

The suggested DyNF-Net, for the case of an m-input-single-output system, comprises generalized Takagi-Sugeno-Kang [6] rules in the form

$$\begin{aligned}&\mathit{IF}\ u_1(k)\ \text{is}\ A_1\ \text{AND}\ u_2(k)\ \text{is}\ A_2\ \text{AND} \ldots \text{AND}\ u_m(k)\ \text{is}\ A_m\\ &\mathit{THEN}\ g(k) = g(\boldsymbol{u}(k))\end{aligned} \tag{1}$$

where $\boldsymbol{u}(k) = [u_1, u_2, ..., u_m]^T$ is the input vector. The rule output $g(\boldsymbol{u}(k))$ is implemented by a recurrent neural network in the form of 1-H-1, having a linear input layer while the hidden and output layers consist of neurons with internal feedback. In particular, the DyNF-Net has the following structural characteristics:

The premise part is static. The degree of fulfillment is the algebraic product of the corresponding membership functions (Gaussian functions):

$$\mu_j(k) = \prod_{i=1}^{m} \mu_{A_{ji}}\left(u_i(k)\right) = \prod_{i=1}^{m} \exp\{-\frac{1}{2}\cdot\frac{(u_i - m_{ji})^2}{(\sigma_{ji})^2}\} \tag{2}$$

The system's overall output is static as well, derived via the weighted average defuzzification method, as given below:

$$y(k) = \sum_{j=1}^{r} \mu_j(k)\cdot g_j(k) \Bigg/ \sum_{j=1}^{r} \mu_j(k) \tag{3}$$

The consequent parts of the r fuzzy rules are dynamic. Their structural elements are neurons with local output feedback at the hidden and output layers. Thus, dynamics is introduced to the overall system through these feedback connections. No feedback connections of the rule's total output or connections among neurons of the same layer exist. The operation of the consequent part of the j-th fuzzy rule is described by the following set of equations:

$$O_{ji}(k) = f_1\left(\sum_{l=1}^{m}(w_{jil}^{(1)} \cdot u_l(k)) + \sum_{l=1}^{D_o}(w_{jil}^{(2)} \cdot O_{ji}(k-l)) + w_{ji}^{(3)}\right) \tag{4}$$

$$g_j(k) = f_2\left(\sum_{i=1}^{H}(w_{ji}^{(4)} \cdot O_{ji}(k)) + \sum_{l=1}^{D_g}(w_{jl}^{(5)} \cdot g_j(k-l)) + w_j^{(6)}\right) \tag{5}$$

where i=1,…,H and j=1,…,r. The following notation is used:

- f_1 and f_2 are the neuron activation functions of the hidden and the output layers, respectively. In the following, the activation functions are both chosen to be the hyperbolic tangent *tanh(.)*.
- $O_{ji}(k)$ is the output of the i-th hidden neuron of the j-th fuzzy rule, at time k, and $g_j(k)$ is the output of the j-th fuzzy rule.
- D_0 and D_g are the time lag orders of the local output feedback, respectively, for the neurons of the hidden and the output layer, respectively.
- $w_{jil}^{(1)}$, $w_{jil}^{(2)}$, $w_{ji}^{(4)}$, $w_{ji}^{(5)}$ are the synaptic weights at the hidden and output layers.
- $w_{ji}^{(3)}$, $w_j^{(6)}$ are bias terms for hidden neurons and the output neuron, respectively.

The formation of the consequent part of the fuzzy rules is given in Fig. 1.
The selection of the aforementioned features is dictated by the following:

- The DyNF-Net preserves the local learning characteristics of the classic TSK model, since it comprises fuzzily interconnected subsystems, which are local-recurrent-global-feedforward neural networks [7]. The rules are not linked with each other in time, neither through external nor internal feedback. They are connected merely via the defuzzification part. The premise part performs the input space partition and the consequent part performs the input-output mapping. Accordingly, each recurrent neural network is capable of tracking the dynamics of the internal states of the unknown system, in the input space's domain that is set by the respective premise part.
- The selection of the particular neuron as structural unit is based on the DFNN model [8], where a more complicated form, nevertheless sharing the same underlying philosophy, was employed and exhibited improved identification characteristics compared to dynamic elements that have local synapse feedback.

4 The Training Algorithm

The DyNF-Net is trained by means of the dynamic resilient back-propagation algorithm (D-RPROP) [9], which constitutes a modification of the standard RPROP method [10], applicable to fuzzy models whose consequent parts are recurrent neural networks. Only minor modifications are made, such that the method takes into consideration the special features of the DyNF-Net, requiring calculation of the error gradients for the feedback weights, as well as for the parameters of the membership functions. A detailed description of the algorithm can be found in [9].

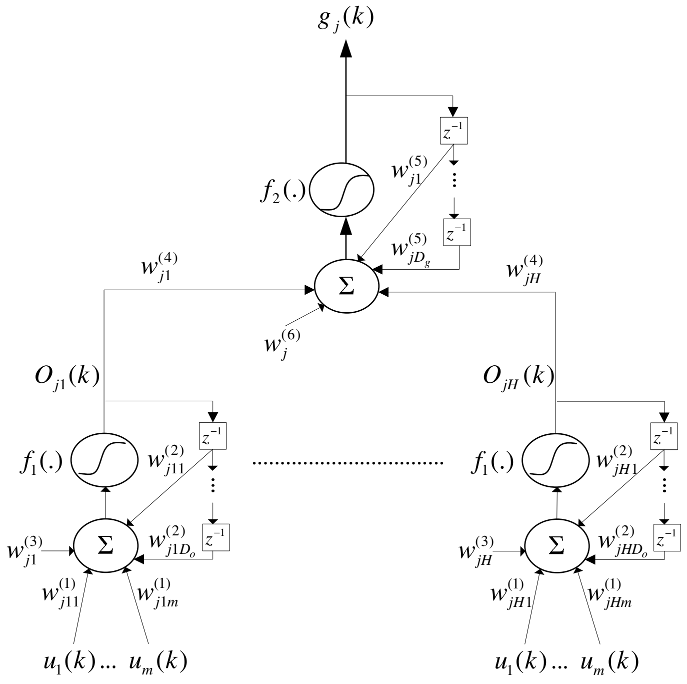


Fig. 1. Consequent part of the *j*-th fuzzy rule

A necessary part of the training algorithm is the extraction of the error gradients. Since the premise and defuzzification parts are static, the gradients of the error function, *E*, with respect to the weights of the premise part, are derived by use of standard partial derivatives, as given in [6]. The gradients of *E* with respect to the weights of the consequent part should be calculated using ordered partial derivatives [11], since temporal relations exist through the feedback connections. Calculation of the error gradients is based on the use of Lagrange multipliers, as shown below:

$$\frac{\partial^+ E}{\partial w_{jil}^{(1)}}, \frac{\partial^+ E}{\partial w_{jil}^{(2)}}, \frac{\partial^+ E}{\partial w_{ji}^{(3)}} = f\left(\lambda_{ji}^{(O)}\right) \tag{6}$$

$$\frac{\partial^+ E}{\partial w_{ji}^{(4)}}, \frac{\partial^+ E}{\partial w_{ji}^{(5)}}, \frac{\partial^+ E}{\partial w_{j}^{(6)}} = f\left(\lambda_{j}^{(g)}\right) \tag{7}$$

with the Lagrange multipliers derived as follows:

$$\lambda_{ji}^{(O)}(k) = \sum_{\substack{l=1 \\ k+l \le k_f}}^{D_o} \left\{ \lambda_{ji}^{(O)}(k+l) \cdot f_1'(k+l, j, i) \cdot w_{jil}^{(2)} \right\} + \lambda_j^{(g)}(k) \cdot f_2'(k, j) \cdot w_{ji}^{(4)} \tag{8}$$

$$\lambda_j^{(g)}(k) = \frac{2}{k_f} \cdot \left(y(k) - y_d(k)\right) \cdot \frac{\mu_j(k)}{\sum_{i=1}^{n} \mu_i(k)} + \\ + \sum_{\substack{l=1 \\ k+l \le k_f}}^{D_g} \left\{ \lambda_j^{(g)}(k+l) \cdot f_2'(k+l,j) \cdot w_{jl}^{(5)} \right\} \tag{9}$$

where $f_2'(k+l,j)$ and $f_1'(k+l,j,i)$ are the derivatives of $g_j(k+l)$ and $O_{ji}(k+l)$, respectively, with respect to their arguments. Equations (8) and (9) are backward difference equations that can be solved for $k = k_f$-1,…1 using the boundary conditions:

$$\lambda_j^{(g)}(k_f) = 2 \cdot \left(y(k_f) - y_d(k_f)\right) \cdot \mu_j(k_f) \Big/ (k_f \cdot \sum_{i=1}^{n} \mu_i(k_f)) \tag{10}$$

$$\lambda_{ji}^{(O)}(k_f) = \lambda_j^{(g)}(k_f) \cdot f_2'(k_f, j) \cdot w_{ji}^{(4)} \tag{11}$$

5 Simulation Results

The data in hand cover a period of 10 years, January 1998 to December 2007, and are the monthly outgoing calls to mobile destinations originating from the PBX of a large organization. The data set is divided into two subsets. The training set, which is used to estimate the parameters associated with each method, and the validation set, which is used for the evaluation of the forecasts. The training set is chosen to be 9 years (108 months) long and the validation set 1 year (12 months) long.

From the visual observation of the time series (Fig. 2) a distinct seasonal pattern is noticeable which is made prevalent from the minimum that occurs in August. Apart from that, the number of calls to mobile destinations shows an increasing trend which comports with reports on mobile services penetration.

The parameters, which are estimated during the fitting procedure, are used to forecast future values of the series. Since the validation set is not used in model fitting, these forecasts are genuine forecasts, and can be used to evaluate the forecasting ability of each model. The forecasting accuracy can be evaluated by means of the accuracy measures mentioned in Section 2.

The fuzzy models for the outgoing calls are decided to be single-input-single-output, with the input being the number of the mobile calls of the previous month, in order to investigate whether the models are able to discover the temporal dependencies of this time-series through its recurrent structure alone. Several DyNF-Nets with different structural characteristics are examined and various combinations of the learning parameters are tested. Selection of the model- parameter combination is based on the criteria of (a) effective identification of the time series and (b) moderate complexity of the resulting models. The selected structural characteristics of the DyNF-Nets are given in Table 1.

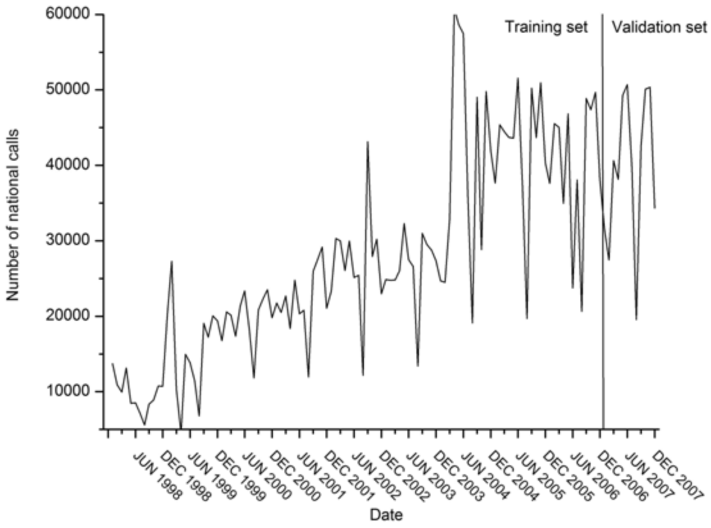


Fig. 2. Monthly number of outgoing calls to mobile destinations

Table 1. Structural characteristics and learning parameters of the DyNF-Net

DyNF-Net structural characteristics				
Kind of calls	Rules	H	D_0	D_g
Mobile	2	6	2	2
D-RPROP Learning Parameters				
Premise part				
n+	n-	Δmin	Δmax	Δ0
1.01	0.95	1E-4	0.30	0.02
Consequent part				
n+	n-	Δmin	Δmax	Δ0
1.10	0.80	1E-4	0.90	0.02

The training process is carried out in parallel mode and lasts for 1500 epochs. The learning parameters of D-RPROP are hosted in Table 1. The input space is normalized to [-1,1] and the initial membership functions are uniformly distributed.

For each method, three holdout accuracy measures were computed. These are the RMSE, the MAPE, and the Theil's U-statistic. The smaller value of each statistic indicates the better fit of the method to the observed data. The results for each one of the twelve models are presented in Table 2; bold numbers indicate best fit.

To further exploit the forecasting ability of the methods, plots of the observed values for the validation set with the best fit model (DyNF-Net) and the second best fit model (Damped MS) for each case are drawn (Fig. 3). Also, 95% prediction intervals (PI) for the forecasts are presented in the plots. The prediction (or confidence) intervals were estimated during the fitting process of the second best model and are denoted as Upper Confidence Level (UCL) and Lower Confidence Level (LCL).

Table 2. Comparative performance evaluation (testing data set) analysis

Model	*RMSE*	*MAPE*	*Theil's U*
DyNF-Net	**5742**	**13.166**	**0.326**
NF1	12009	28.875	1.000
LESA-M	9915	23.046	0.747
LESA-ADD	10271	27.218	0.699
SES	9671	24.698	0.569
Holt's Linear	11191	35.507	0.663
Winter's MS	9114	20.475	0.665
Winter's AS	8495	21.875	0.573
Damped NoS	11962	31.756	0.715
Damped MS	7419	15.958	0.524
Damped AS	9020	23.584	0.599
SARIMA	10102	20.793	0.775

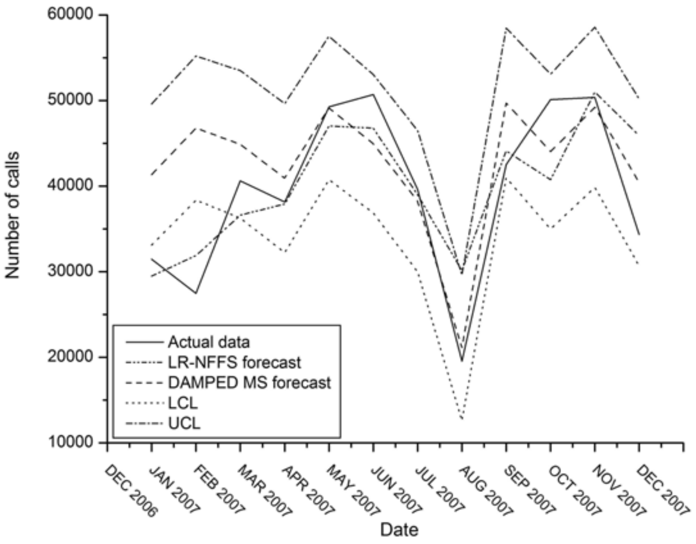


Fig. 3. Forecast fit with 95% PI and a comparison of DyNF-Net with the second best fit method (Damped MS) for each data set

Visual observation of the plot (Fig. 3) reveals the differences between the two best-fit models. DyNF-Net gives better forecast, as it follows the evolution of the series more closely, identifies the first local minimum that appears in February, but misses the significance of the minimum in August. The second best-fit model (indicated by all three statistics in Table 2) is a damped trend model with multiplicative seasonality (Damped MS). One may observe how the DyNF-Net forecast follows the actual data pattern against the second best model, which misses the behavior of the first two months (January and February 2007). Moreover, the actual data are not even within the 95% prediction interval of the Damped MS forecast.

Interestingly, the second best-fit model (Damped MS) was the best fit model indicated for the same type of calls in a past analysis [1] and was attributed to “the high cost of mobile calls, which desists users from making many calls to mobile destinations and retards the upward tendency”. With our current point of view an alternative reasoning for this damped trend may be the approach to a saturation point in mobile penetration. It should be also pointed out that a recent review of forecasting in operational research [12] concluded that the damped trend can “reasonably claim to be a benchmark forecasting method for all others to beat” which was the case with our DyNF-Net approach for the mobile data.

References

1. Hilas, C.S., Goudos, S.K., Sahalos, J.N.: Seasonal decomposition and forecasting of telecommunication data: A comparative case study. Technological Forecasting & Social Change 73, 495–509 (2006)
2. Madden, G., Joachim, T.: Forecasting telecommunications data with linear models. Telecommunications Policy 31, 31–44 (2007)
3. Makridakis, S.G., Wheelwright, S.C., McGee, V.E.: Forecasting: Methods and applications, 3rd edn. Wiley, New York (1998)
4. Gardner Jr., E.S.: Exponential smoothing: The state of the art. Journal of Forecasting 4, 1–28 (1985)
5. Box, G.E.P., Jenkins, G.M.: Time series analysis: Forecasting and control, 2nd edn. Holden-Day, San Francisco (1976)
6. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. IEEE Transactions on Systems, Man, and Cybernetics 15, 116–132 (1986)
7. Tsoi, A.C., Back, A.D.: Locally recurrent globally feedforward networks: A critical review of architectures. IEEE Transactions on Neural Networks 5, 229–239 (1994)
8. Mastorocostas, P.A., Theocharis, J.B.: A Recurrent Fuzzy Neural Model for Dynamic System Identification. IEEE Transactions on Systems, man, and Cybernetics – Part B 32, 176–190 (2002)
9. Mastorocostas, P.A.: Resilient back propagation learning algorithm for recurrent fuzzy neural networks. IET Electronics Letters 40, 57–58 (2004)
10. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: Proceedings of IEEE International Joint Conference on Neural Networks, pp. 586–591 (1993)
11. Piche, S.W.: Steepest descent algorithms for neural network controllers and filters. IEEE Transactions on Neural Networks 5, 198–212 (1994)
12. Fildes, R., Nikolopoulos, K., Crone, S., Syntetos, A.: Forecasting and operational research: a review. Journal of the Operational Research Society 59, 1150–1172 (2008)

Evolutionary Learning of Regularization Networks with Multi-kernel Units

Petra Vidnerová and Roman Neruda

Institute of Computer Science
Academy of Sciences of the Czech Republic,
Pod vodárenskou věží 2, Praha 8, Czech Republic
petra@cs.cas.cz

Abstract. Regularization networks represent an important supervised learning method applicable for regression and classification tasks. They benefit from very good theoretical background, although the presence of meta parameters is their drawback. The meta parameters, including the type of kernel function, are typically supposed to be given in advance and come ready as an input of the algorithm. In this paper, we propose multi-kernel functions, namely product kernel functions and composite kernel functions. The choice of kernel function becomes part of the optimization process, for which a new evolutionary learning algorithm is introduced that deals with different kernel functions, including composite kernels. The results are demonstrated on experiments with benchmark tasks.

1 Introduction

Regularization networks (RN) benefit from very good theoretical background, since the regularization theory presents a sound framework for solving supervised learning problems [1–3]. Moreover, there exists a simple, yet quite efficient learning algorithm introduced in [4]. A disadvantage of regularization network approach is the presence of meta parameters that are supposed to be known in advance. These meta parameters are the type of kernel function and the regularization parameter. In addition, the kernel function typically has additional parameters, for instance the width of the Gaussian kernel.

It is known from the theory that we can consider networks with multi-kernel units, although there have not been many case of applying such an architecture in practice. Multi-kernel units are known to possess the same theoretical properties as the single kernel units, yet, they can represent the underlying geometry of the data better.

In this paper we introduce a method for optimization of RN meta parameters. The method is based on minimization of cross-validation error, which is an estimate of generalization ability of a network, by means of genetic algorithms. Different species are utilized corresponding to different kinds of kernel functions, thus a natural co-evolution is employed to solve the meta-learning process. The algorithm is also able to represent multi-kernel functions mentioned above.

The paper is organized as follows. In the next section, we introduce the regularization network. In Section 3, sum and composite kernel functions are introduced. Section 4 describes the genetic parameter search. Section 5 contains results of our experiments. Conclusion can be found in section 6.

© Springer-Verlag Berlin Heidelberg 2011

2 Regularization Networks

In order to develop regularization networks we formulate the problem of supervised learning as a function approximation problem. 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 is ill-posed, we have to add some *a priori* knowledge about the function f. We usually assume that the function is *smooth*, in the sense that two similar inputs correspond to two similar outputs, and that 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 approach has sound 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 [3, 4].

Poggio and Smale in [4] proposed a learning algorithm (Alg. 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:

$$\min_{f\in\mathcal{H}_K} H[f], \text{where } H[f] = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\boldsymbol{x}_i))^2 + \gamma||f||^2_K. \tag{2}$$

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 Alg. 1 is in its simplicity and effectiveness. However, network needs fixed values of *meta parameters* such as γ, the type of kernel function, and its parameters (e.g. a width of the Gaussian). Then, the algorithm reduces to the problem of solving linear system of equations (4).

The real performance of the algorithm depends significantly on the choice of meta parameters γ and kernel function. However, their optimal choice depends on a particular data set and there is no general heuristic for setting them.

```
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

       $(N\gamma I + K)\boldsymbol{w} = \boldsymbol{y}$,    (4)

   where $I$ is the identity matrix, $K_{i,j} = K(\boldsymbol{x_i}, \boldsymbol{x_j})$,
   and $\boldsymbol{y} = (y_1, \ldots, y_N)$, $\gamma > 0$.
```

Algorithm 1. RN learning algorithm

3 Composite and Product Kernel Functions

In this section we will introduce composite and product types of kernel functions. A special case of composite kernel function — sum kernels — has been studied in our previous work [5], while product kernel functions were first introduced in [6, 7].

The kernel function, used in the RN learning algorithm Alg. 1, is traditionally supposed to be given in advance, for instance chosen by a user. In fact, the choice of a kernel function is equivalent to the choice of a prior assumption about the problem at hand. Therefore it seems that such a choice is crucial for the quality of the solution and should be always done according to the given task. However, the choice of the kernel is most often not part of the learning algorithm.

The real data are often heterogeneous. The heterogeneity refers either to attributes or parts of the input space, or both. By the former we mean that different attributes are of different types or differ in quality. By the latter that the data have different qualities (such as density) in different parts of the input space. Then for different parts of data different kernel functions are suitable.

We believe that, in such situations, kernel functions that are created as a combination of simpler kernel functions might better reflect the character of data. Therefore we proposed multi-kernel functions, namely a product kernel and a composite kernel.

By a *composite kernel* we mean a kernel function K that can be expressed as a linear combination of other kernel functions $K(\boldsymbol{x}, \boldsymbol{y}) = \alpha K_1(\boldsymbol{x}, \boldsymbol{y}) + \beta K_2(\boldsymbol{x}, \boldsymbol{y})$, where K_1 and K_2 are kernel functions, $\alpha, \beta \in \mathbb{R}$.

By a *product kernel* we mean a kernel function K that can be expressed as a product of other kernel functions $K((\boldsymbol{x}_1, \boldsymbol{x}_2), (\boldsymbol{y}_1, \boldsymbol{y}_2)) = K_1(\boldsymbol{x}_1, \boldsymbol{y}_1) K_2(\boldsymbol{x}_2, \boldsymbol{y}_2)$, where K_1 and K_2 are kernel functions.

We can combine different kernel functions or combine two kernel functions of same type but with different parameters such as two Gaussians of different widths (note that the Gaussians have a same a center).

4 Genetic Parameter Search

In our approach we use genetic algorithms (GA) [8] that represent a sound and robust technique used to find approximate solutions to optimization and search problems. The use of GA gives us versatility, thus with suitable solution encoding, we can search for different type of kernel units, including product and composite kernels within the framework of one algorithm. The genetic algorithms typically work with a population of *individuals* embodying abstract representations of feasible solutions. Each individual is assigned a *fitness* that is a measure of how good solution it represents. The better the solution, the higher the fitness value.

The population evolves towards better solutions. The evolution starts from a population of completely random individuals and iterates in generations. In each generation, the fitness of each individual is evaluated. Individuals are stochastically selected from the current population (based on their fitness), and modified by means of operators *mutation* and *crossover* to form a new population. The new population is then used in the next iteration of the algorithm.

4.1 Encoding

We work with individuals encoding the parameters of RN learning algorithm (Alg. 1). They are the type of kernel function, its additional parameters, and the regularization parameter. When the type of the kernel function is known in advance, the individual consists only of the kernel's parameter (i.e. the width in case of Gaussian kernel) and the regularization parameter.

In case of simple kernel function, the individual is encoded as

$$I = \{\text{type of kernel function}\ ,\text{kernel parameter}\ ,\gamma\},$$

i.e. $I = \{\text{Gaussian}\ , \text{width} = 0.5, \gamma = 0.01\}$.

In case of composite kernel function, the individual looks like

$$\begin{aligned} I = \{\ & \alpha, \text{type of kernel function } K_1, \text{kernel parameter}, \\ & \beta, \text{type of kernel function } K_2, \text{kernel parameter}, \gamma\}. \end{aligned}$$

Product kernels are encoded as

$$\begin{aligned} I = \{\ & \text{list of attributes included in } x_1, \text{type of kernel function } K_1, \text{kernel parameter}, \\ & \text{list of attributes included in } x_2, \text{type of kernel function } K_2, \text{kernel parameter}, \gamma\}. \end{aligned}$$

4.2 Operators

New generations of individuals are created using operators *selection*, *crossover* and *mutation*. Mutation operator is implemented as a standard biased mutation introducing small random perturbation to numerical values of existing individuals (by adding a small random float drawn from normal distribution).

To work with individuals representing different kernel types we introduce the co-evolution principle of species. Individuals with different kernel functions naturally represent different species, where each specie forms one subpopulation. The selection is performed on the whole population and the selected individual is inserted into subpopulation according its kernel type. Crossover is then performed only among the individuals of same subpopulation.

The crossover for individuals representing simple kernels of the same type operates as a version of arithmetic crossover where kernel parameter and γ are subject to the operator. In our case, the new parameters are chosen randomly from the interval formed by the old values. In case of sum and composite kernels, the crossover works as a standard interchange of sub-kernels.

4.3 Fitness

The optimal RN should not only approximate the data from the training set, but also has a good generalization ability. Our estimate of a generalization ability is a *cross-validation error*. Then, it can be stated that we search for such meta parameters that optimize the cross-validation error. Since we want to minimize the cross-validation error, the fitness should reflect it. So the lower the cross-validation error is, the higher the fitness value is.

See Alg. 2 for the sketch of the algorithm.

```
Input: Data set $S = \{\boldsymbol{x}_i, y_i\}_{i=1}^N \subseteq \mathbb{R}^n \times \mathbb{R}$
Output: Parameters $\gamma$ and $K$.

1. Create randomly an initial population $P^0 = P^0_{K_1} \cup \ldots \cup P^0_{K_n}$,
   where $K_i$ is
   a particular kernel function and each $P^0_{K_i}$ has $M$
   individuals $I^0_{K_i,1} \ldots I^0_{K_i,M}$.
2. $i \leftarrow 0$
3. $\forall j : P^{i+1}_{K_j} \leftarrow$ empty set
4. for $j = 0$ to $n * M$:
  (a) $I \leftarrow selection(P^i)$
  (b) insert $I$ into $P^{i+1}_K$ such that $I = \{K, p, \gamma\}$
5. for $j = 0$ to $n$:
  (a) for $k = 1$ to $\frac{|P^{i+1}_{K_j}|}{2}$:
        with probability $p_{cross}$:
        $(I^{i+1}_{K_j,2*k}, I^{i+1}_{K_j,2*k+1}) \leftarrow crossover(I^{i+1}_{K_j,2*k}, I^{i+1}_{K_j,2*k+1})$
  (b) for $k = 1$ to $|P^{i+1}_{K_j}|$:
        with probability $p_{mutate}$: $I^{i+1}_{K_j,k} \leftarrow mutate(I^{i+1}_{K_j,k})$
6. $\forall I \in P^{i+1}$:
  (a) $E \leftarrow E_{cross}(\gamma, K, p, S)$, where $I = \{K, p, \gamma\}$
  (b) fitness($I$) $\leftarrow$ C - E
7. $P^{i+1} \leftarrow P^{i+1}_{K_1} \cup \ldots \cup P^{i+1}_{K_n}$
8. $i \leftarrow i + 1$
9. goto 3 and iterate until the fitness stops increasing
```

Algorithm 2. Genetic parameter search

5 Experiments

For our experiments we used the collection of benchmark problems *Proben1* introduced in [9]. The tasks are listed in Table 1. Each task is present in three variants corresponding to three different partitioning to training and testing sets.

The following procedure is used for experiments:

1. find the values for γ and K using genetic parameter search
2. use the whole training set and the parameters found by Step 1 to estimate the weights of RN
3. evaluate error on the testing set

The error is computed as $E = 100\frac{1}{Nm}\sum_{i=1}^{N}||\boldsymbol{y}_i - f(\boldsymbol{x}_i)||^2$, where $||\cdot||$ denotes the Euclidean norm.

The standard numerical library LAPACK [10] was used to solve linear systems.

Table 1. Overview of Proben1 tasks. Number of inputs (n), number of outputs (m), number of samples in training and testing sets (N_{train},N_{test}). Type of task: approximation or classification.

Task name	n	m	N_{train}	N_{test}	**Type**
cancer	9	2	525	174	class
card	51	2	518	172	class
flare	24	3	800	266	approx
glass	9	6	161	53	class
heartac	35	1	228	75	approx
hearta	35	1	690	230	approx
heartc	35	2	228	75	class
heart	35	2	690	230	class
horse	58	3	273	91	class

Three experiments were performed. In the first one we evolved elementary kernel functions using genetic algorithm with species. The population had 4 subpopulation, corresponding to the four common kernel functions: Gaussian ($K(x,y) = e^{-||x-y||^2}$), multiquadric ($K(x,y) = (||x-y||^2 + c^2)^{-1/2}$), inverse multiquadric ($K(x,y) = (||x-y||^2 + c^2)^{1/2}$), and sigmoid ($K(x,y) = tanh(xy - \theta)$).

In the second experiment, the product kernels were evolved, while in the third experiment we evolved composite kernels. The population in these two latter cases had 50 individuals. Note that the regularization parameter as well as kernel parameters were also found by evolution. The evolution algorithm was run for 300 generations.

Table 2 lists training and testing errors for networks found by genetic parameter search, first with the Gaussian kernel (which is the most commonly used kernel function), then by the inverse multiquadric (which was a winning kernel function in all cases of genetic search for optimal elementary function), and finally by product and composite kernels.

The Gaussian function gives the best test errors only in four cases out of 27, despite the fact that it is the most commonly used kernel. The inverse multiquadric function, that was the absolute winner among elementary kernel functions, gives best test errors in 14 cases. Product kernels give best test error in 9 cases, composite kernels in 10 cases.

Inverse multiquadric		Composite Kernel	
E_{train}	E_{test}	E_{train}	E_{test}
1.83	**1.50**	0.14	1.53

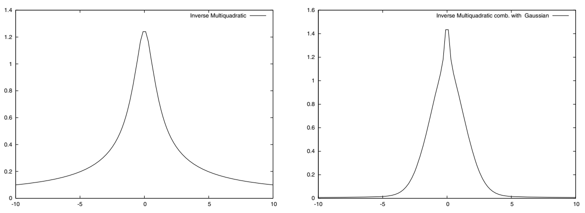


Fig. 1. Winning kernels for **cancer1** task

Inverse multiquadric		Composite Kernel	
E_{train}	E_{test}	E_{train}	E_{test}
1.41	**2.92**	1.34	**2.92**

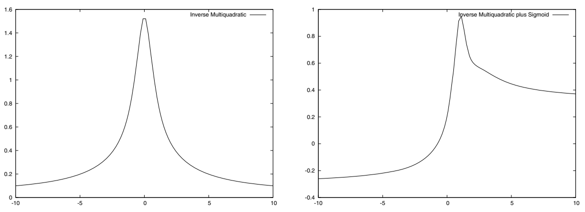


Fig. 2. Winning kernels for **cancer2** task

Inverse multiquadric		Composite Kernel	
E_{train}	E_{test}	E_{train}	E_{test}
2.32	6.13	2.19	**6.05**

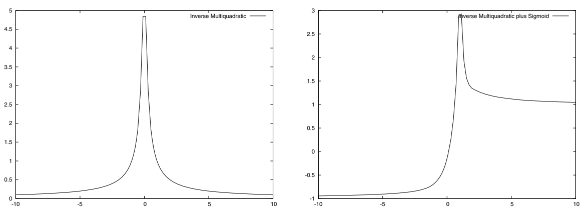


Fig. 3. Winning kernels for **glass1** task

In terms of training errors, the Gaussian function gives best results in 10 cases, inverse multiquadric function in 1 case and composite kernels in 20 cases.

In some cases, composite kernels achieved very low training errors while preserving test errors comparable to other kernels. In this cases we get very precise approximation

Table 2. Error on training and testing set obtained by RN with Gaussian kernels, inverse multi-quadric kernels, product and composite kernels

	Gaussian kernel		Inv. multiquadric		Product kernel		Composite kernel	
Task	E_{train}	E_{test}	E_{train}	E_{test}	E_{train}	E_{test}	E_{train}	E_{test}
cancer1	2.29	1.76	1.83	**1.50**	2.68	1.81	0.14	1.53
cancer2	1.85	3.01	1.41	**2.92**	2.07	3.61	1.34	**2.92**
cancer3	2.05	2.78	1.74	**2.54**	2.28	2.81	0.51	2.85
card1	7.17	10.29	7.56	10.03	9.22	**9.99**	6.45	10.06
card2	5.72	13.19	6.06	**12.74**	7.96	12.90	4.60	**12.74**
card3	5.93	12.68	6.35	12.28	6.94	**12.23**	6.35	12.29
flare1	0.34	**0.54**	0.35	**0.54**	0.36	**0.54**	0.35	**0.54**
flare2	0.41	**0.27**	0.41	**0.27**	0.42	0.28	0.41	**0.27**
flare3	0.39	**0.33**	0.39	**0.33**	0.40	0.34	0.39	**0.33**
glass1	3.37	6.99	2.32	6.13	2.64	7.31	2.19	**6.05**
glass2	3.92	7.74	1.06	**6.79**	2.55	7.46	0.76	6.85
glass3	4.11	7.36	2.67	**6.27**	3.31	7.26	0.50	6.61
heartac1	3.39	3.30	3.66	3.03	4.22	**2.76**	3.71	3.06
heartac2	2.08	4.15	2.42	3.95	3.49	**3.87**	2.40	3.95
heartac3	2.52	**5.10**	2.85	5.13	3.26	5.18	0.64	**5.10**
hearta1	2.58	4.43	2.73	**4.28**	3.47	4.39	2.49	**4.28**
hearta2	2.12	4.32	2.41	**4.20**	3.28	4.29	2.46	4.22
hearta3	2.59	4.44	2.84	**4.40**	3.40	4.44	1.44	4.41
heartc1	7.73	16.03	8.23	15.93	10.00	16.08	1.39	**15.59**
heartc2	10.67	6.82	10.99	6.47	12.37	**6.29**	0.02	6.38
heartc3	6.88	13.36	7.22	12.86	8.71	**12.65**	7.23	12.85
heart1	8.56	13.70	9.21	**13.55**	9.56	13.67	0.87	13.76
heart2	7.99	14.15	8.17	13.88	9.43	**13.86**	8.22	13.92
heart3	5.85	17.03	5.92	16.85	9.15	**16.06**	5.85	16.88
horse1	2.52	13.31	4.15	**11.77**	14.25	12.45	0.26	11.93
horse2	1.58	16.12	3.63	15.22	12.24	15.97	0.72	**15.03**
horse3	2.27	14.62	3.84	13.53	9.63	15.88	0.33	**13.22**

of training data and still we have good generalization ability. Most of these kernels are combinations of wide kernel and narrow kernel. The narrow member ensures the precise answer for the training point, the wider kernel is responsible for generalization.

For illustration, the simple kernel functions and composite kernel functions evolved for the cancer1, cancer2 and glass1 tasks are presented in Fig. 1, Fig. 2 and Fig. 3, respectively.

Time complexity of the evolution is quite high, in terms of hours for tasks considered in this work. However, for the larger data sets the kernel function and the other meta-parameters can be found on a subsection of the data set and only the single last training of the RN is performed on the whole data.

6 Conclusion

In this paper we have explored the possibilities of using multi-kernel units for regularization networks. While the subject is theoretically sound, there has not been a suitable learning algorithm that makes use of these properties so far. We have proposed an evolutionary learning algorithm that considers multi-kernel units and adjusts their parameters in order to capture the geometrical properties of training data better. The learning process has been tested on several benchmark tasks with promising results. In general, the multi-kernels have demonstrated superior performance in comparison to single kernel units. The most common winning architecture consists of the combination of a wide and narrow kernel function.

Although only kernels defined on real numbers are considered throughout this work, in practical applications we often meet data containing attributes of different types, such as enumerations, sets, strings, texts, etc. Such data may of course be converted to real numbers by suitable preprocessing. But the regularization network learning framework may be generalized to be able to work on such data types. For such generalization sophisticated kernel functions defined on various types were introduced in literature. Examples of kernel functions defined on objects including graphs, sets, texts, etc. can be found in [11].

Acknowledgments

This research has been supported by the Grant Agency of the Czech Republic under project no. P202/11/1368, and by the Institutional Research Plan AV0Z10300504 “Computer Science for the Information Society: Models, Algorithms, Applications".

References

1. Girosi, F., Jones, M., Poggio, T.: Regularization theory and Neural Networks architectures. Neural Computation 2, 219–269 (1995)
2. Kůrková, V.: Learning from data as an inverse problem. In: Antoch, J. (ed.) Computational Statistics, pp. 1377–1384. Physica Verlag, Heidelberg (2004)
3. Poggio, T., Girosi, F.: A theory of networks for approximation and learning. Technical report, Cambridge, MA, USA (1989); A. I. Memo No. 1140, C.B.I.P. Paper No. 31
4. Poggio, T., Smale, S.: The mathematics of learning: Dealing with data. Notices of the AMS 50, 536–544 (2003)
5. Neruda, R., Vidnerová, P.: Genetic algorithm with species for regularization network metalearning. In: Papasratorn, B., Lavangnananda, K., Chutimaskul, W., Vanijja, V. (eds.) IAIT 2010. CCIS, vol. 114, pp. 192–201. Springer, Heidelberg (2010)
6. Kudová, P., Šámalová, T.: Product kernel regularization networks. In: Ribeiro, B., Albrecht, R.F., Dobnikar, A., Pearson, D.W., Steele, N.C. (eds.) Adaptive and Natural Computing Algorithms, pp. 433–436. Springer, Wien (2005)
7. Kudová, P., Šámalová, T.: Sum and product kernel regularization networks. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 56–65. Springer, Heidelberg (2006)
8. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1996)
9. Prechelt, L.: PROBEN1 – a set of benchmarks and benchmarking rules for neural network training algorithms. Technical Report 21/94, Universitaet Karlsruhe (September 1994)
10. LAPACK: Linear algebra package, `http://www.netlib.org/lapack/`
11. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)

Solving the Assignment Problem with the Improved Dual Neural Network*

Xiaolin Hu[1] and Jun Wang[2]

[1] State Key Laboratory of Intelligent Technology and Systems,
Tsinghua National Laboratory for Information Science and Technology (TNList),
Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, China

[2] Department of Mechanical and Automation Engineering,
The Chinese University of Hong Kong, Shatin,
N.T., Hong Kong, China

Abstract. During the last two decades, several neural networks have been proposed for solving the assignment problem, and most of them either consist of $O(n^2)$ neurons (processing units) or contain some time varying parameters. In the paper, based on the improved dual neural network proposed recently, we present a new assignment network with 2n neurons and some constant parameters only. Compared with the existing neural networks for solving the assignment problem, its more favorable for implementation. Numerical simulation results indicate that the time complexity of the network is $O(n)$.

Keywords: Assignment problem, sorting problem, quadratic programming, linear programming, analog circuits.

1 Introduction

The assignment problem is concerned with assigning n entities to n slots for achieving minimum cost or maximum profit. It is known to be a polynomial combinatorial optimization problem. Its applications cover pattern classification, machine learning, operations research and so on.

For solving the assignment problem, there exist many efficient iterative algorithms such as the auction methods [1], signature methods [2]. Inspired by the Hopfield network for solving optimization problems with neural networks, many continuous methods were developed for solving the assignment problem (e.g., [3–7]). One of the major advantages of this kind of methods is that they can be

* X. Hu's work was supported by the National Natural Science Foundation of China under Grant Nos. 60805023 and 90820305, National Basic Research Program (973 Program) of China under Grant no. 2007CB311003, and Basic Research Foundation of Tsinghua National Laboratory for Information Science and Technology (TNList). J. Wang's work was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grants CUHK417209E and CUHK417608E.

© Springer-Verlag Berlin Heidelberg 2011

implemented in parallel analog circuits to achieve super high speed, which is very attractive in real-time applications. However, most of these methods requires an "annealing" procedure which is sensitive to the solution quality. In addition, this time-varying procedure would pose difficulties in circuits implementation.

In [8], an assignment neural network is proposed without any time-varying procedure. It features elegant theoretical results with constant parameters. The major demerit lies in its complex structure. For instance, it entails more neurons and interconnections than the network in [7]. But this is a tradeoff between the efficiency and structural complexity, as argued in [8]. In the present paper, we show that this tradeoff is unnecessary. Based on the *improved dual neural network* or IDNN proposed in [9], we have designed a very simple assignment network with constant parameters only.

2 Problem Formulation

Suppose that there are n entities to be assigned to n slots and assigning entity i to slot j induces cost c_{ij}. Then, what is the best assignment in terms of minimum total cost? This is called *assignment problem.* Mathematically, it can be formulated as a zero-one programming problem:

$$\begin{array}{rl} \text{minimize} & f_1 = \sum_{i=1}^n \sum_{j=1}^n c_{ij}x_{ij} \\ \text{subject to} & \sum_{i=1}^n x_{ij} = 1 \quad \forall j = 1,\ldots,n \\ & \sum_{j=1}^n x_{ij} = 1 \quad \forall i = 1,\ldots,n \\ & x_{ij} \in \{0,1\} \quad \forall i,j = 1,\ldots,n \end{array} \tag{1}$$

where c_{ij} and x_{ij} are, respectively, cost variable and decision variable associated with assigning entity i to slot j. The variable $x_{ij} = 1$ means assigning entity i to slot j and $x_{ij} = 0$ means not assigning entity i to slot j. Since any entity should be, and must be, assigned to only one slot, and any slot should be, and must be, assigned one entity, a feasible assignment should correspond to a matrix $\mathbf{x} = \{x_{ij}\}$ with only one element equal to one in every column and row.

Lemma 1. *The problem* (1) *is equivalent to the following problem*

$$\begin{array}{rl} \textit{minimize} & f_2 = \frac{q}{2} \sum_{i=1}^n \sum_{j=1}^n x_{ij}^2 + \sum_{i=1}^n \sum_{j=1}^n c_{ij}x_{ij} \\ \textit{subject to} & \sum_{i=1}^n x_{ij} = 1 \quad \forall j = 1,\ldots,n \\ & \sum_{j=1}^n x_{ij} = 1 \quad \forall i = 1,\ldots,n \\ & x_{ij} \in \{0,1\} \quad \forall i,j = 1,\ldots,n, \end{array} \tag{2}$$

where $q > 0$ *is a constant.*

Proof. This can be proved by showing that $\sum_{i=1}^n \sum_{j=1}^n x_{ij}^2$ is a constant in the feasible region. Because $x_{ij} \in \{0,1\}$, $x_{ij} = x_{ij}^2$. Then

$$\sum_{i=1}^n \sum_{j=1}^n x_{ij}^2 = \sum_{i=1}^n \sum_{j=1}^n x_{ij} = n,$$

which completes the proof.

Problem (1) and problem (2) are zero-one programming problems. But, if the problem has a unique solution, it is known that (1) is equivalent to the following (continuous) linear programming problem

$$\begin{array}{rl} \text{minimize} & f_1 = \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \\ \text{subject to} & \sum_{i=1}^n x_{ij} = 1 \quad \forall j = 1, \ldots, n \\ & \sum_{j=1}^n x_{ij} = 1 \quad \forall i = 1, \ldots, n \\ & x_{ij} \in [0,1] \quad \forall i, j = 1, \ldots, n. \end{array} \tag{3}$$

In what follows, we formulate the continuous counterpart of problem (2).

Theorem 1. *If the problem (2) has a unique solution, then there exists a sufficiently small positive constant q such that (2) is equivalent to the following problem*

$$\begin{array}{rl} \textit{minimize} & f_2 = \frac{q}{2} \sum_{i=1}^n \sum_{j=1}^n x_{ij}^2 + \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \\ \textit{subject to} & \sum_{i=1}^n x_{ij} = 1 \quad \forall j = 1, \ldots, n \\ & \sum_{j=1}^n x_{ij} = 1 \quad \forall i = 1, \ldots, n \\ & x_{ij} \in [0,1] \quad \forall i, j = 1, \ldots, n. \end{array} \tag{4}$$

Proof. What is only needed to show is that the unique solution of (2), denoted by $\mathbf{x}^*$, is also a solution of (4), by considering that the objective function of (4) is strictly convex and it has at most one solution.

Denote the feasible region of (4) by $\mathcal{X}$ and the feasible region of (2) by $\mathcal{V}$. Note that any point in $\mathcal{X}$ is called a *doubly stochastic matrix* and any point in $\mathcal{V}$ is called a *permutation matrix*. According to the well-known Birkhoff-von Neumann theorem, a square matrix is doubly stochastic if and only if it is a convex combination of permutation matrices. Namely, any $\mathbf{x} \in \mathcal{X}$ can be expressed as

$$\mathbf{x} = \sum_k \theta_k \mathbf{v}^{(k)},$$

where $\mathbf{v}^{(k)} \in \mathcal{V}$, $\sum_k \theta_k = 1, 0 \leq \theta_k \leq 1$. Denote the unique solution of (2) by $\mathbf{x}^*$. Then

$$\begin{aligned} \langle \mathbf{x}^*, \mathbf{x}^* - \mathbf{v}^{(k)} \rangle &= \sum_{i=1}^n \sum_{j=1}^n x_{ij}^* (x_{ij}^* - v_{ij}^{(k)}) = \sum_{i=1}^n \sum_{j=1}^n (x_{ij}^*)^2 - x_{ij}^* v_{ij}^{(k)} \\ &= n - \sum_{i=1}^n \sum_{j=1}^n x_{ij}^* v_{ij}^{(k)} > 0 \end{aligned}$$

for $\mathbf{v}^{(k)} \in \mathcal{V}, \mathbf{v}^{(k)} \neq \mathbf{x}^*$, where $\langle \cdot, \cdot \rangle$ stands for the Frobenius inner product of two matrices. Because $\mathbf{x}^*$ is also the unique solution of (3), according to the equivalence between convex optimization problem and variational inequality [10], we have

$$\langle \nabla f_1(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle = \langle \mathbf{c}, \mathbf{x} - \mathbf{x}^* \rangle \geq 0 \quad \forall \mathbf{x} \in \mathcal{X}.$$

Now we show that for any $\mathbf{x} \in \mathcal{X}$ but $\mathbf{x} \neq \mathbf{x}^*$, the above equality cannot hold. Otherwise, there exists such a point $\bar{\mathbf{x}}$ so that $\langle \mathbf{c}, \bar{\mathbf{x}} - \mathbf{x}^* \rangle = 0$. Then

$\langle \mathbf{c}, \mathbf{x} - \mathbf{x}^* \rangle = \langle \mathbf{c}, \mathbf{x} - \bar{\mathbf{x}} \rangle \geq 0, \forall \mathbf{x} \in \mathcal{X}$, indicating that $\bar{\mathbf{x}} \neq \mathbf{x}^*$ is also a solution of problem (3), which contradicts to the uniqueness of the solution. Therefore

$$\langle \mathbf{c}, \mathbf{x} - \mathbf{x}^* \rangle > 0 \quad \forall \mathbf{x} \in \mathcal{X}, \mathbf{x} \neq \mathbf{x}^*.$$

Consequently,

$$\langle \mathbf{c}, \mathbf{v}^{(k)} - \mathbf{x}^* \rangle > 0 \quad \forall \mathbf{v}^{(k)} \in \mathcal{V}, \mathbf{v}^{(k)} \neq \mathbf{x}^*,$$

as $\mathcal{V} \subset \mathcal{X}$. Let

$$0 < q \leq \frac{\langle \mathbf{c}, \mathbf{v}^{(k)} - \mathbf{x}^* \rangle}{\langle \mathbf{x}^*, \mathbf{x}^* - \mathbf{v}^{(k)} \rangle}, \quad \forall \mathbf{v}^{(k)} \in \mathcal{V}, \mathbf{v}^{(k)} \neq \mathbf{x}^*.$$

It follows that

$$\langle \nabla f_2(\mathbf{x}^*), \mathbf{v}^{(k)} - \mathbf{x}^* \rangle = \langle q\mathbf{x}^* + \mathbf{c}, \mathbf{v}^{(k)} - \mathbf{x}^* \rangle \geq 0, \quad \forall \mathbf{v}^{(k)} \in \mathcal{V}, \mathbf{v}^{(k)} \neq \mathbf{x}^*.$$

For any $\mathbf{x} \in \mathcal{X}$,

$$\langle \nabla f_2(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle = \langle \nabla f_2(\mathbf{x}^*), \sum_k \theta_k \mathbf{v}^{(k)} - \mathbf{x}^* \rangle = \sum_k \theta_k \langle \nabla f_2(\mathbf{x}^*), \mathbf{v}^{(k)} - \mathbf{x}^* \rangle \geq 0,$$

where $0 \leq \theta_k \leq 1, \sum_k \theta_k = 1$. Hence, $\mathbf{x}^*$ is a solution of (4), which completes the proof.

From the proof, it can be seen that q should be small enough and its upper bound is

$$\min_k \frac{\langle \mathbf{c}, \mathbf{v}^{(k)} - \mathbf{x}^* \rangle}{\langle \mathbf{x}^*, \mathbf{x}^* - \mathbf{v}^{(k)} \rangle} \tag{5}$$

which is positive. In practice, the optimal solution $\mathbf{x}^*$ is unknown, and it is time consuming to find all feasible solutions $\mathbf{v}^{(k)}$ to problem (1) or (2). So, it is suggested to set q to be very small. Throughout the rest of the paper, it is assumed that the solution of the assignment problem (1) is unique. Then, with small q, all of the four problems (1), (2), (3), (4) are equivalent.

3 The Improved Dual Neural Network

3.1 Architecture

The problem (4) is a quadratic programming problem. According to [9], it can be solved by using the improved dual neural network (IDNN):

– state equations

$$\begin{cases} \tau \dfrac{du_i}{dt} = -\displaystyle\sum_{j=1}^{n} g(u_i + v_j - c_{ij}/q) + 1, \quad i = 1, \dots, n \\ \tau \dfrac{dv_i}{dt} = -\displaystyle\sum_{j=1}^{n} g(u_j + v_i - c_{ji}/q) + 1, \quad i = 1, \dots, n \end{cases} \tag{6a}$$

– output equations

$$x_{ij} = g(u_i + v_j - c_{ij}), \quad i, j = 1, \ldots, n, \tag{6b}$$

where $\tau > 0$ is the time constant and

$$g(s) = \begin{cases} 0, \, s < 0, \\ s, \, 0 \leq s \leq 1, \\ 1, \, s > 1. \end{cases}$$

Clearly, this network would entail n neurons for representing u_i and n neurons for representing v_i. The block diagram of the neuron u_i is plotted in Fig. 1. Neurons corresponding to the dual variables v_i can be constructed similarly.

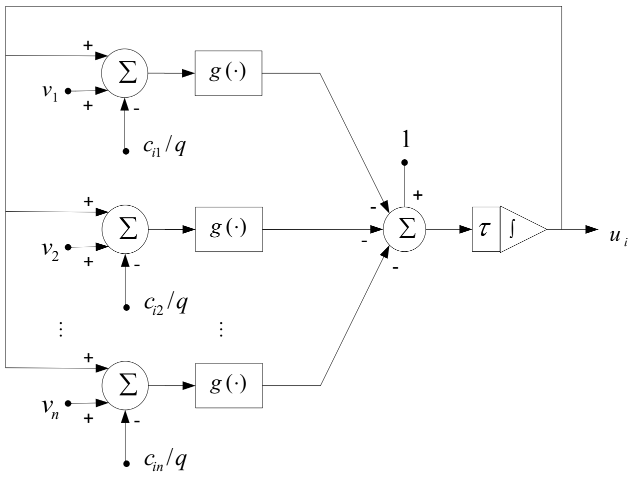


Fig. 1. Block diagram for realizing neuron u_i

3.2 Comparison with Other Neural Networks

In [7], two neural networks were proposed for solving the assignment problem (1). One is called the *primal neural network*, whose dynamic equations are

– state equations

$$\tau \frac{du_{ij}}{dt} = -\sum_{k=1}^{n}(x_{ik} + x_{kj}) + 2 - \alpha c_{ij} \exp\left(-\frac{t}{T}\right), \quad i, j = 1, \ldots, n \tag{7a}$$

– output equations

$$x_{ij} = h(u_{ij}), \quad i, j = 1, \ldots, n, \tag{7b}$$

where $\tau > 0$ is the time constant, $\alpha > 0$ and $T > 0$ are scaling constants, and h is a nonnegative and monotone nondecreasing activation function satisfying

$g(z) \geq 0$ and $dg/dz \geq 0$ for any $z \in \mathcal{R}$ (a typical example is the sigmoid function). The other model is called the *dual neural network*, whose dynamic equations are

– state equations

$$\begin{cases} \tau \dfrac{du_i}{dt} = -\displaystyle\sum_{j=1}^{n} \phi(u_i + v_j - c_{ij}) + \alpha \exp\left(-\dfrac{t}{T}\right), & i = 1, \ldots, n \\ \tau \dfrac{dv_i}{dt} = -\displaystyle\sum_{j=1}^{n} \phi(u_j + v_i - c_{ji}) + \alpha \exp\left(-\dfrac{t}{T}\right), & i = 1, \ldots, n \end{cases} \tag{8a}$$

– output equations

$$x_{ij} = \delta(u_i + v_j - c_{ij}), \quad i, j = 1, \ldots, n, \tag{8b}$$

where τ, α, T are positive constant, ϕ is a nonnegative and nondecreasing function defined as $\phi(z) = 0$ if $z \leq 0$ and $\phi(z) > 0$ if $z > 0$, and δ is a function defined as $\delta(z) = 1$ if $z = 0$, or $\delta(z) = 0$ otherwise. It is seen that the primal neural network entails n^2 neurons and $O(n^2)$ connections whereas the dual neural network entails $2n$ neurons and $O(n^2)$ connections (see [7] for details). The latter model is simpler in structure. In addition, from (8), it can be seen that the dual neural network is nearly as simple as the IDNN (6).

The two models above were derived from the *deterministic annealing neural network* for solving the general convex optimization problems [11]. So, both of them involve a temperature parameter $\alpha \exp(-t/T)$, which is hard to be realized in hardware.

In [8], a neural network called *primal-dual neural network* was proposed for solving the assignment problem. The state equations are

$$\begin{cases} \tau \dfrac{dx_{ij}}{dt} = -\left(c_{ij} \displaystyle\sum_{p=1}^{n}\sum_{q=1}^{n}(c_{pq}x_{pq} - u_p - v_p) + \phi(-x_{ij}) + \sum_{l=1}^{n}(x_{il} + x_{lj}) - 2\right), \\ \hfill i, j = 1, \ldots, n \\ \tau \dfrac{du_i}{dt} = -\left(-\displaystyle\sum_{p=1}^{n}\sum_{q=1}^{n}(c_{pq}x_{pq} - u_p - v_p) + \sum_{l=1}^{n}\phi(u_i + v_l - c_{il})\right), \ i = 1, \ldots, n \\ \tau \dfrac{dv_i}{dt} = -\left(-\displaystyle\sum_{p=1}^{n}\sum_{q=1}^{n}(c_{pq}x_{pq} - u_p - v_p) + \sum_{l=1}^{n}\phi(u_l + v_i - c_{li})\right), \ i = 1, \ldots, n \end{cases} \tag{9}$$

where $\tau > 0$ and ϕ is defined the same as in (8). This network entails constant parameters only, which is superior to the networks (7) and (8). However, it is more complicated in architecture because it consists of $n^2 + 2n$ neurons and $O(n^2)$ connections.

Since the assignment problem can be equivalently written as a linear programming problem or quadratic programming problem, all linear programming networks and quadratic programming networks are capable of solving it. Currently, there are two main streams of researches in this area, characterized by continuous or discontinuous activation functions. Some typical networks in the former refer to [12–14]. However, it is easy to see that all of them require $n^2 + 2n$ neurons for solving the assignment problem, though most of them are relatively simpler than the primal-dual network (9). Some typical networks in the latter refer to [15, 16]. It is easy to see that they require at least n^2 neurons for this problem.

3.3 Stability and Convergence

The following result follows from Theorem 2 in [9] and Theorem 1, directly.

Lemma 2. *Let* (u_i^*, v_i^*) *be the equilibrium point of* (6)*, then the unique solution of the assignment problem* (1) x_{ij}^* *can be denoted by* $g(u_i^* + v_j^* - c_{ij})$*.*

Since it is assumed that the problem (4) has a unique solution, from Theorem 4 in [9] and Lemma 2, it follows the following theorem.

Theorem 2. *If* q *is sufficiently small, any equilibrium point of the network* (6) *is stable in the sense of Lyapunov, and the corresponding output trajectory globally converges to the unique solution of the assignment problem* (1)*.*

4 Numerical Simulations

To verify the correctness of the results presented in last section, we have numerically simulated the proposed network (6) for solving some problems in MATLAB with the “ode23” function.

We randomly generated some cost matrices $\mathbf{c}$ with every element between zero and one. Fig. 2 shows the state trajectories of the network (6) for solving such a problem with $n = 10$. The parameters were set as follows: $\tau = 10^{-6}$ and $q = 0.001$. The output of the network converged to the correct solution of (1), verified by solving the linear programming problem with the MATLAB build-in function “linprog”.

According to Theorem 1, q should be set sufficiently small. But it is unclear how small is enough and what is the influence of the values of q on convergence properties. We investigated this issue numerically. The idea is to compare the convergence time of the network with different q values. Let $\mathbf{x}^*$ be the *ground truth* obtained by MATLAB function “linprog”. For saving CPU time in running simulations, we terminated the program when

$$\sum_i \sum_j |\mathrm{round}(x_{ij}(t)) - x_{ij}^*| \leq 10^{-6}$$

where round(x) is equal to 0 if $x < 0.5$ and 1 otherwise. The time at which this was achieved was recorded as the convergence time. For every q value, 30 different runs with random initial points between $[-50, 50]$ were executed. The

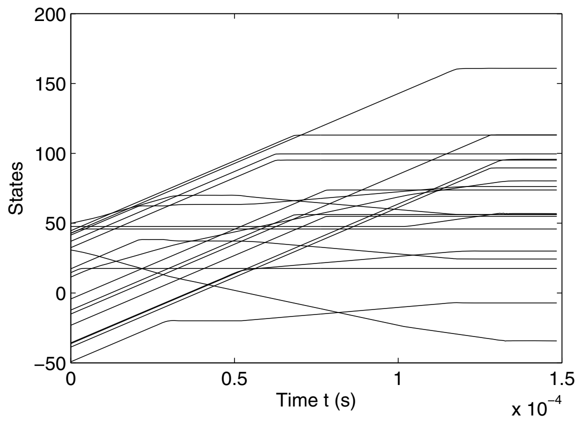
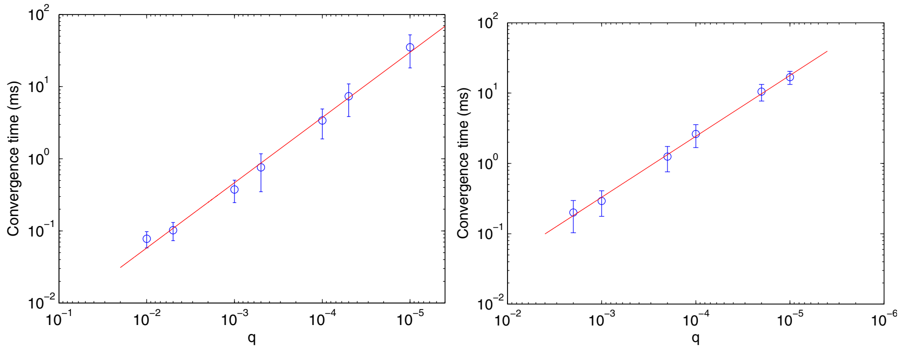


Fig. 2. State trajectories of the network ([6])



(a) (b)

Fig. 3. Convergence time of the network ([6]) with different q values for (a) $n = 10$ and (b) $n = 50$. Note that the coordinates are in logarithm scale. The circles denote the mean and the bars below and above them stand for one standard deviation each. The continuous line in each plot is the linear regression of the logarithm of the mean versus the logarithm of q.

statistics of the convergence time for two problem sizes is plotted in Fig. [3]. It is seen that as q decreases the convergence time increases, and the logarithm of the convergence time is roughly a linear function of the logarithm of q. In other words, the convergence time increases as q decreases. Therefore, q should not be too small. This poses some difficulty in choosing an appropriate q.

Next, we investigated the relationship between the problem size and the convergence time of the network. Let $q = 0.01/n$. For each $n = 10, 20, 30, 40, 50, 60, 70, 80, 100$, thirty different runs with random initial points between [-50,50] were executed. The same stopping criterion as above was adopted. The statistics of

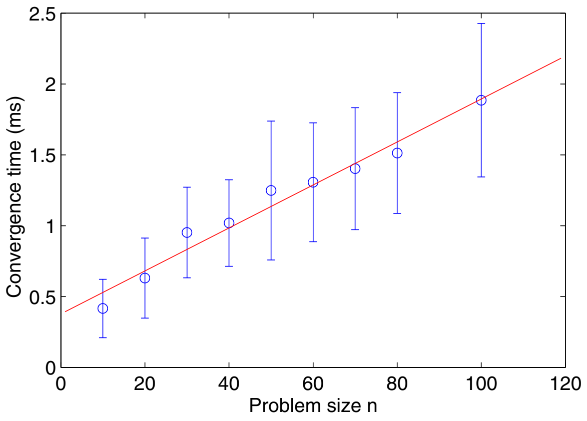


Fig. 4. Convergence time of the network (6) for different problem sizes. The circles and bars denote the mean and the standard deviation, respectively. The continuous line is the linear fit of the means versus n.

the convergence time is plotted in Fig. 4. It is seen that the convergence time grows linearly with the problem size, which is in sharp contrast with the iterative algorithms such as [2] which has $O(n^3)$ time complexity.

5 Concluding Remarks

We applied the *improved dual neural network* or IDNN for solving the assignment problem. An assignment network with $2n$ neurons, free of time-varying parameters, was obtained, which was theoretically guaranteed to be globally convergent to the solution of the problem if only the solution is unique. Numerical simulations indicated that the computing time was roughly a linear function of the problem size n, implying that this network method, if implemented in hardware, would be much faster than traditional iterative algorithms for solving large scale problems.

The main disadvantage of the new assignment network is that it introduces a parameter that should be set appropriately. If it is too large, the network may converge to incorrect states; if it is too small, the convergence will be slow. In most cases, it has to be selected by trial and error.

References

1. Bertsekas, D.P.: A new algorithm for the assignment problem. Math. Prog. 21(1), 152–171 (1981)
2. Balinski: Signature methods for the assignment problem. Operations Research 33(3), 527–536 (1985)
3. Eberhardt, S.P., Daud, T., Kerns, D.A., Brown, T.X., Thakoor, A.P.: Competitive neural architecture for hardware solution to the assignment problem. Neural Networks 4, 431–442 (1991)

4. Wang, J.: Analogue neural network for solving the assignment problem. Electronics Letters 28(11), 1047–1050 (1992)
5. Kosowsky, J.J., Yuille, A.L.: The invisible hand algorithm: solving the assignment problem with statistical physics. Neural Networks 7(3), 477–490 (1994)
6. Urahama, K.: Analog circuit for solving assignment problem. IEEE Trans. Circuits Syst. I 40, 426–429 (1994)
7. Wang, J.: Primal and dual assignment networks. IEEE Trans. Neural Netw. 8(3), 784–790 (1997)
8. Wang, J., Xia, Y.: Analysis and design of primal-dual assignment networks. IEEE Trans. Neural Netw. 9(1), 183–194 (1998)
9. 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 Netw. 19(12), 2022–2031 (2008)
10. Kinderlehrer, D., Stampcchia, G.: An Introduction to Variational Inequalities and Their Applications. Academic, New York (1980)
11. Wang, J.: A deterministic annealing neural network for convex programming. Neural Networks 7(4), 629–641 (1994)
12. Hu, X., Zhang, B.: A new recurrent neural network for solving convex quadratic programming problems with an application to the k-winners-take-all problem. IEEE Trans. Neural Netw. 20(4), 654–664 (2009)
13. Hu, X., Zhang, B.: An alternative recurrent neural network for solving variational inequalities and related optimization problems. IEEE Trans. Syst., Man, Cybern. B 39(6), 1640–1645 (2009)
14. Cheng, L., Hou, Z.G., Tan, M.: A delayed projection neural network for solving linear variational inequalities. IEEE Trans. Neural Netw. 20(6), 915–925 (2009)
15. Forti, M., Nistri, P., Quincampoix, M.: Generalized neural network for nonsmooth nonlinear programming problems. IEEE Trans. Circuits Syst. I 51(9), 1741–1754 (2004)
16. Liu, Q., Wang, J.: A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming. IEEE Trans. Neural Netw. 19(4), 558–570 (2008)

Evaluating Model on Effectiveness of Network Defense Missile Based on LMBP Neural Network

Cenrui Ma, Guangzheng Long, and Yun Yang

The Missile Institute of Air Force Engineering
University Xi an, China
mcrwztg@163.com

Abstract. For effectively evaluating the operational effectiveness of the netting air-defense missile system, the index system of the operational effectiveness for the netting air-defense missile system is studied, and evaluating index is uniformly quantized according to utility function. And sets up utility function equation for the netting air-defense missile system, studies the operational effectiveness evaluating model of the netting air-defense missile system based on LMBP neural network, then it is validated that the model has higher value by an example.

Keywords: netting air-defense missile system; operational effectiveness; LM Algorithm; LMBP neural network.

1 Introduction

The netting air-defense missile system warfare is a combined operation that includes multiple type defense missile under the uniform command, which emphasizes the comprehensive operational effectiveness by uniformly commanding and controlling the netting air-defense missile to effectively develop various type air-defense missile. In fact, the netting air-defense missile system warfare is a special synergetic warfare, which is made up of the four stages: information acquiring, operation decision-making optimizing, intercepting actions and operational evaluation feeding back according to the fighting sequence. And The netting air-defense missile system warfare effectiveness is influenced by the multiple factors, whose relation is very complex, always have the different direction and content of the operational effectiveness, and these relations cannot be distinguished when the different factors are simultaneously working. So a specific mathematical model showing their relations is not easily obtained. NN(neural network), whose large scale parallel computing, distributed processing capacity, self-organization, adaptive ability and self-learning capacity, can be easily used to process these problems that need to apply the multiple factors and conditions meanwhile, therefore, for effectively evaluating the operational effectiveness of the netting air-defense missile system, the evaluating model of the netting air-defense missile system based on LMBP NN is set up[1-2].

© Springer-Verlag Berlin Heidelberg 2011

2 Evaluating Index System of the Operational Effectiveness for the Netting Air-Defense Missile System

The evaluating index system of the operational effectiveness for the netting air-defense missile system should be given according to its detection capacity, command and decision-making, attack and kill rate, operational ability and adaptive capacity. The evaluating index system of the operational effectiveness for the netting air-defense missile system is obtained after consulting opinion of many experts and summarizing the references[3] in the following figure 1.

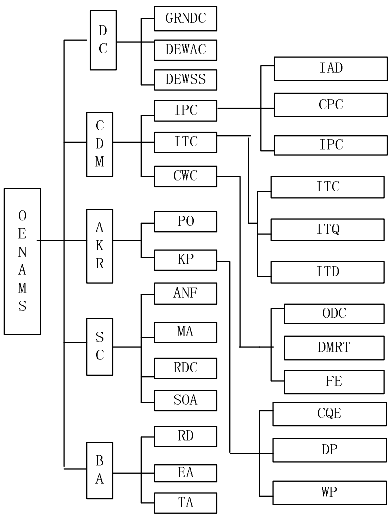


Fig. 1. Evaluating index system of the operational effectiveness

OENAMS- the operational effectiveness of the netting air-defense missile system, DC-detection capacity, CDM-command and decision-making, AKR-attack and kill rate, SC-survival capacity, BA-battlefield adaptability, GRNDC-ground radar net detection capacity, DEWAC- detection and early warning of air scout, DEWSS-detection and early warning of spy satellite, IPC-information processing capacity, ITC- information transmission capacity, CWC- command and warfare capacity, PO-probability of the object that can be fired, KP-kill probability, ANF-the ability that can not be found, MA-maneuvering ability, RDC-resist damage ability, SOA-system organizing again, EA-environment adaptation, TA-tactics adaptation, RD- resist disturbance, IAD- information acquiring density, CPC-comprehensive processing delay, IPC- information processing capacity, ITC- information transmission capacity, ITQ- information transmission quality, ITD- information transmission delay, ODC-object distribution capacity, DMRT-decision-making respond time, FE-fire efficiency, CQE-control and guide error, DPD-detonating probability, WP-warhead power.

3 Evaluating Model on the Operational Capacity of the Netting Air-Defense Missile System Based on LMBP NN

3.1 Neural Net and LMBP Algorithm

BPNN is a multi-level feeding back net, which uses a learning algorithm of training of network. The comprehensive evaluation method based on LMBP NN, which utilizes the self-learning, adaptive capacity, fast evaluation and powerful fault tolerance, approaches the human thinking, which integrates quantitative and qualitative method. And it avoids the uncertainty and the subjective influence from the anthropic weighting and the related coefficient because the weight vale of the model is obtained by an example.

The principle of LMBP NN includes is that the output value acquiring under the activation function according to the initial joints value and the threshold value of the network with the input of some samples. And compares the actual output value with the expected output value, when it has some error, BPNN can be back propagatation from the output value, continually adjusts the joint and the threshold value and make it smaller and smaller that the RMS error between the actual value and the expectation, when the error gets to the precision demand, shows that the BPNN has been trained and can be applied.

BPNN is made up of the input layer, the hidden layer and the output layer. Activation function usually applies the nonlinear S type function in the following equation (1).

$$f(x) = \frac{1}{1+e^{-x}} \tag{1}$$

so a three-layer BPNN is set up in this paper, whose model is as the following figure 2.

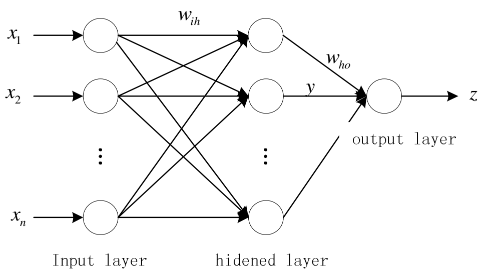


Fig. 2. Three-layer BPNN model

The procedure of the typical BP learning algorithm includes:

1. set the initial weight value, threshold value and the learning cell of BPNN, and so on, then put forward the net precision request and initialize the network.
2. input the training sample, train the network until it satisfy the precision request.

3. calculate the error between the actual and the expected output value according to the given input value, if not satisfy the precision request, then go on the step 4, otherwise return the step 2.

4. correct the weight value and threshold value according to the calculation result of the same layer unit, and return the step 2.

The algorithm and procedure of the LMBP includes the followings.

NN may be trained when its weight value and the threshold value are initialized. The training algorithm of BPNN makes the effectiveness function value the smallest by calculating the effectiveness function gradient, adjusting the weight value and the threshold value according to the negative gradient direction. The regulation function formula can be expressed as the following equation 2 in the k cycle.

$$x_{k+1} = x_k - \alpha_k g_k \tag{2}$$

x_k —the current weight value and the threshold value;

g_k —the current performance function gradient;

α_k —the study speed.

LM optimization algorithm, which is also called the damping least square method, regulates its weight value according to the following equation 3.

$$\Delta w = [\boldsymbol{J}^T \boldsymbol{J} + u\boldsymbol{I}]^{-1} \boldsymbol{J}^T e \tag{3}$$

e —the error vector;

$\boldsymbol{J}$ —Jacobian matrix that the error differentializes the weight value;

u —one quantity.

Therefore, LM algorithm is a method that smoothly reconciles between the steepest descent and Gauss-newton method. And its specific procedure is the followings.

1. distil all the input signals into the net and calculate the output signals of the net, simultaneously train the quadratic sum of the error all the objects that is calculated by the error function.

2. calculate the Jacobian matrix that the error differentializes the weight value, first, define the Marquardt sensitivity:

$$S_i^m = \frac{\partial E}{\partial n_i^m} \tag{4}$$

from the equation 4, it is well known that the error function E expresses the sensitivity of the change of the i cell from the m layer, n is the weight sum of every layer.

The Marquardt sensitivity recurrence relation equation is the following formula.

$$S_q^m = E(n_q^m)(w^{m+1})^T S_q^{m+1} \tag{5}$$

Therefore, it can be back propagation from the last layer to the first layer, as follows the formula 6.

$$S^m \to S^{m-1} \to \cdots \to S^2 \to S^1 \tag{6}$$

Then calculate the element of Jacobian matrix by the equation 7, as follows the formula 7.

$$[J]_{h,l} = \frac{\partial e_{k,q}}{\partial w_{i,j}^m} = \frac{\partial e_{k,q}}{\partial n_{i,q}^m} \frac{\partial e_{i,q}^m}{\partial w_{i,j}^m} = S_{i,h}^m \frac{\partial n_{i,q}^m}{\partial w_{i,j}^m} = S_{i,h}^m a_{j,q}^{m-1} \tag{7}$$

3. calculate the Δw by equation 3.

4. repeatedly calculate the quadratic sum of the error by $w + \Delta w$. If the new sum is smaller than the sum in the equation 1, then divide $\theta(\theta > 1)$ by using the μ, and return the step 1; otherwise divide $\theta(\theta > 1)$ by using the μ and return the step 3. When the quadratic of the error to some objective error, the algorithm can be considered converge.

One of the keys of the BP algorithm is to choose the number of the hidden layer, if the number of net is not enough, the degree of freedom of the net is not enough,learning for expansion is not enough as well, and the fault tolerance is bad; contrarily the degree of freedom of the net is excessive, the learning time will extend and the error is not certainly good. The following formula may be referred[1][9].

$$h = \sqrt{m \cdot n} \tag{8}$$

$$\sum_{i=1}^{n} C_h^i > p \tag{9}$$

$$h \leq \sqrt{m(n+3)+1} \tag{10}$$

h -the neural cell number of the hidden layer
n - the neural cell number of the input layer
m - the neural cell number of the output layer
p -the sum of the samples

These formulas should be synthetically applied in actual application, and the best neural cell number of the hidden layer is chosen by combining the actual training result in the computer.

3.2 Evaluating Model Based on LMBP NN

The procedure of setting up the evaluating model is the followings.

1. normalization of the initial data.

BPNN working function demands that the input signals should be the real number in the scope [0,1], but the various parameters in the index system have different unit systems, and the main parameter value is all over this scope. So it is necessary that these parameters should be normalized. And the its method is as follows. Suppose that $\min x_k$ is the smallest value of k index, $\max x_k$ is the largest value of k index, x_k is the initial training sample value.

Cost-type index can be disposed by the following formula (11).

$$x_k' = \frac{\max x_k - x_k}{\max x_k - \min x_k} \tag{11}$$

Benefit-type index can be disposed by the following formula (12).

$$x_k' = \frac{x_k - \min x_k}{\max x_k - \min x_k} \tag{12}$$

Qualitative index subordination can be obtained by the confidence distribution method. Suppose that some expert gives some index value x_j, whose confidence level is a_j, then (x_j, a_j), $j = 1, 2, \cdots, n$ can be obtained. So,qualitative index subordination can be obtained by the following formula (13).

$$R(j) = \frac{\sum_{j=1}^{n}(a_j x_j)}{\sum_{j=1}^{n} a_j} \tag{13}$$

2. setting up the neural net model
the typical three layer BPNN is made up of the input layer, the hidden layer and the output layer. Therefore the input layer is the evaluation index system of the netting air-defense missile warfare effectiveness. the hidden layer distinguishes the relation between the input layer cell and he output layer cell, and extracts the different character that the input layer influences the output layer. The output layer is an expression of the operational effectiveness for the netting air-defense missile system. Then the 23×5×1 net structure model has been made[4-5].

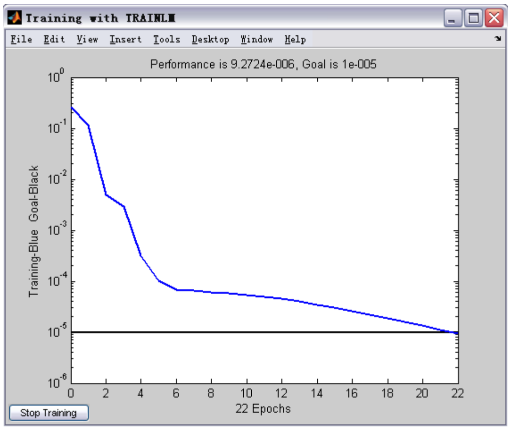
4 Analysis of a Real Example

The BPNN can realize the given input and output relation because it can study and adjust the connection weight value of the net by using the study samples, therefore the training sample acquiring is important, the sample not only demands to include the scope of the max and min value but also contains the median value. The sample data of this example is made up of the expert'evaluation value and the simulation data from t the netting air-defense missile that is in attack and defense. Therefore it makes the sample more rational. The training sample can be obtained by statistic data and the expert'experiences in the following table 1.

Table 1. Training sample

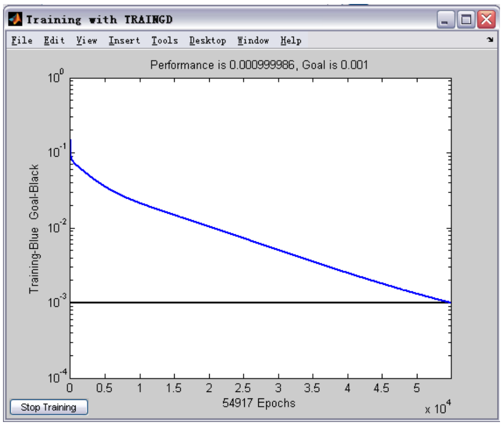
sample	Evaluation index		
1 0.9 0.9 0.90.90.9		0.9 0.90.90.90.90.90.90.90.9	0.90.90.90.90.90.90.9
2 0.8 0.8 0.80.80.8		0.8 0.80.80.80.80.80.80.80.8	0.8 0.80.80.80.80.80.8.
3 0.6 060.60.60.6		0.6 060.60.60.60.6 060.60.60.6	0.6 060.60.60.60.6 060.6
4 0.50.50.50.50.5		0.50.50.50.50.50.50.50.50.5	0.50.50.50.50.50.50.5
5 0.40.40.40.40.4		0.40.40.40.40.4 0.40.40.40.40.4	0.40.40.40.40.4 0.40.40.4
6 0.90.70.80.60.6.		0.50.70.80.70.6 0.80.70.90.60.8	0.90.70.50.80.9 0.70.80.6
7 0.70.60.80.70.6		0.40.80.60.40.9 0.70.60.30.60.4	0.80.60.70.60.5 0.70.70.8
8 0.60.50.50.40.7		0.40.60.40.50.8 0.60.70.50.40.6	0.50.40.80.60.60.50.7

LMBPNN application procedure is made by the Matlab, whose algorithm net training process is in the following figure 3, which can be used to evaluate the netting air-defense missile warfare effectiveness.



Performance is 9.2724e-006, Goal is 1e-005

Fig. 3. LM algorithm net training process



Performance is 0.000999986, Goal is 0.001

Fig. 4. Steepest descent net training process

By comparison, it is disposed by the steepest descent method that is usually applied, whose algorithm net training process is in the following figure 4. so the former is more than the steepest descent method.

For validating the evaluating capacity of LMBPNN, some types missile are chosen to evaluate the netting air-defense missile warfare effectiveness, whose index value is 0.7, 0.6, 0.5,0.6,0.7,0.8,0.6,0.5,0.8,0.6,0.7, 0.9,0.5,0.4,0.7,0.9,0.5,0.6,0.5,0.7,0.4,0.8,0.6. the output value is 0.8241, which is obtained by the trained LMBPNN. The result is basically the same as the qualatative evaluation of the experts, and it demonstrates that the model can be effectively applied to evaluate the netting air-defense missile warfare effectiveness.

5 Conclusion

The index system of the operational effectiveness for the netting air-defense missile system is set up according to its operational effectiveness characteristic, and the evaluating method of the netting air-defense missile system based on LMBP neural network is applied to evaluate the netting air-defense missile warfare effectiveness, which in a sense diminishes the human subjective factors. And LM algorithm is applied, which has not the defect of the BP that has the slow converge, so makes the model a higher value. Therefore it is validated that the model and method can be referred for evaluating the netting air-defense missile warfare effectiveness by a real example.

References

1. De Feng, Z.: MATLAB neural net application and design. Mechanic industry press, Beijing (2009) (in Chinese)
2. Xiaohui, G., Bing, C.: The synthetic evaluation of warhead overall efficiency. Journal of Systems Engineering and Electronics 14(1), 12–17 (2003) (in Chinese)
3. Le, W.M., Guang, G.X.: New concepts on the viability of missile system warfare. System and Electric Technology 21(1), 8–10 (1999) (in Chinese)
4. Ping, Y., Ming, B.Y.: the synthetic evaluation model and method on the operational efficiency of the common warhead. Journal of Firepower and Command Control 32(12), 69–72 (2007) (in Chinese)
5. Hui, G.X., Ming, W.X., Shou, Z.Y.: the application of the gray AHP method in the evaluation on the overall efficiency of the intelligent warheadl. Journal of millitary operation and Systems Engineering 2, 51–55 (2002) (in Chinese)

A New Neural Network for Solving Nonlinear Programming Problems

Yun Gao, Xianyun Xu, and Yongqing Yang

School of Science, Jiangnan University,
Wuxi 214122, PR China

Abstract. In this paper a new neural network is proposed to solve nonlinear convex programming problems. The proposed neural network is shown to be asymptotically stable in the sense of Lyapunov. Comparing with the existing neural networks, the proposed neural network has fewer state variables and simpler architecture. Numerical examples show that the proposed network is feasible and efficient.

Keywords: asymptotical stability, neural network, nonlinear programming.

1 Introduction

Nonlinear programming problems are commonly encountered in modern science and technology, such as optimal control, signal processing, pattern recognition[1]. In many engineering applications, the real-time solution of optimization problems is required. However, traditional algorithms may not be efficient since the computing time required for a solution is greatly dependent on the dimension and structure of the problems. One possible and very promising approach to real-time optimization is to apply artificial neural networks[2, 3].

In 1985 Tank and Hopfield first proposed the neural network for linear programming problems [4]. Their work has inspired many researchers to investigate other neural network models for solving programming problems. Over the years, neural network models have been developed to solve nonlinear programming problems. Kennedy and Chua [5] presented a neural network for solving nonlinear programming problems. It is known that the neural network model contains finite penalty parameters and generates approximate solutions only. To avoid using penalty parameters, many other methods have been done in recent years. Zhang and Constantinides developed [6] a Lagrange neural network for a nonlinear programming problem with equality constrains. Xia [7] presented some primal neural networks for solving convex quadratic programming problems and nonlinear convex optimization problems with limit constraints. In order to simplify the architecture of the dual neural network, a simplified dual neural network was introduced for solving convex quadratic programming problems [8]. Leung[9] and Yang[10]presented a feedback neural network for solving convex nonlinear programming problems. Based on the idea of successive approximation, the equilibrium point sequence of subnetworks can converge to an exact optimal solution.

© Springer-Verlag Berlin Heidelberg 2011

In [11, 12], Liu and Wang proposed some one-layer recurrent neural networks for solving linear and quadratic programming problems. The one-layer recurrent neural networks have more simply architecture complexity than the other neural networks such as Lagrangian network and projection network. Recently, several projection neural networks were developed for solving general nonlinear convex programming problems [13–21], which were globally convergent to exact optimal solutions.

In this paper, we present a new neural network for solving nonlinear programming problems. The proposed neural network is asymptotically stable in the sense of Lyapunov. Compared with existing neural networks for such problems, the proposed neural network has fewer state variables, simpler architecture and weaker convergence conditions.

This paper is divided into 5 sections. In section 2, the nonlinear convex programming problem is described. A new neural network model is proposed to solve this problem. In section 3, we prove the stability of the proposed neural network. Two examples are provided to show the effectiveness of the proposed neural network in section 4. Finally, some conclusions are found in section 5.

2 Problem and the Neural Network Model

In this section,we describe the nonlinear convex programming problem and its equivalent formulation. Then we construct a new neural network for solving this problem.

Consider the following nonlinear convex programming problem:

$$\begin{aligned} min \quad & f(x) \\ subject\,to \quad & Ax = b \\ & g(x) \geq 0. \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \cdots, x_n)^T \in R^n$, $g(x) = (g_1(x), g_2(x), \cdots, g_p(x))^T \in R^p$, $f(x)$, $-g_i(x)$ $(i = 1, 2, \cdots, p)$ are continuously differentiable and convex form R^n to R, $A \in R^{m\times n}$, and $rank(A) = m \, (0 < m < n)$, $b \in R^m$.

According to the Karash-Kuhn-Tucker (KKT) conditions for convex optimization [1], the following set of equations has the same solutions as the problem(1).

$$\nabla f(x) + A^T\lambda - g'(x)^T y = 0 \tag{2}$$

$$Ax = b \tag{3}$$

$$g(x) \geq 0, y \geq 0, y^T g(x) = 0 \tag{4}$$

From(2) and (3), we have

$$[I - A^T(AA^T)^{-1}A][\nabla f(x) - g'(x)^T y] + A^T(AA^T)^{-1}(Ax - b) = 0$$

Let $P = A^T(AA^T)^{-1}A$, then the above equation can be written as

$$(I - P)[\nabla f(x) - g'(x)^T y] + Px - Q = 0 \tag{5}$$

where $Q = A^T(AA^T)^{-1}b$.

By the projection theorem , (4) is equivalent to solving the following equation:

$$(y - g(x))^+ - y = 0 \tag{6}$$

where $(y)^+ = ([y_1]^+, \cdots, [y_n]^+)^T$, $[y_i]^+ = max\{0, y_i\}$.

Based on (5) and (6), we propose a neural network for solving (1) as follows:

$$\begin{aligned} \tfrac{dx}{dt} &= -2\{(I - P)[\nabla f(x) - g'(x)^T (y - g(x))^+] - Px + Q\} \\ \tfrac{dy}{dt} &= -y + (y - g(x))^+ \end{aligned} \tag{7}$$

Remark 1. From above analysis, it is obvious that x^* is an optimal solution of (1) if and only if $(x^*, y^*)^T$ is an equilibrium point of system (7).

Remark 2. Comparing with the existing neural networks for solving problem (1). It is easy to see the neural network in [19], has $n + m + p$ state variables. However, our neural network has only $n + p$ state variables.

Remark 3. In [16, 17], Xia proposed recurrent neural networks for nonlinear convex optimization subject to linear or nonlinear constraints. Under the conditions that $f(x)$ or $g_i(x)$ are assumed to be strictly convex, the convergence of the proposed neural network are obtained. However, our neural network can converge to the optimal solution if only $f(x)$ and $-g_i(x)$ are convex.

3 Stability Analysis

In this section, we prove the global asymptotical stability of the neural network(7).

Lemma 1. *Assume $f(x)$ and $-g_i(x)$ are convex. Let (x^*, y^*) be an equilibrium point of neural network (7) and $\varphi(x, y) = f(x) + \frac{1}{2} \parallel (y - g(x))^+ \parallel^2$, then*
(I) $\varphi(x, y)$ is a differential convex function and

$$\nabla \varphi(x, y) = \begin{bmatrix} \nabla f(x) - g'(x)^T (y - g(x))^+ \\ (y - g(x))^+ \end{bmatrix}$$

(II) $\varphi(x, y) - \varphi(x^*, y^*) - (x - x^*)^T (\nabla f(x^*) - g'(x^*)^T y^*) - (y - y^*)^T y^* \geq 0.$

The proof of this Lemma is easy and we omit it.

Theorem 1. *The neural network (7) is asymptotically stable in the sense of Lyapunov for any initial point $(x(t_0), y(t_0)) \in R^{n+p}$.*

Proof. Let $(x^*, y^*)^T$ be an equilibrium of (7), considering the following Lynapov function

$$\begin{aligned} V(x, y) = {} & \varphi(x, y) - \varphi(x^*, y^*) - (x - x^*)^T \nabla \varphi_x(x^*, y^*) - (y - y^*)^T \nabla \varphi_y(x^*, y^*) \\ & + \frac{1}{2}\|x - x^*\|^2 + \frac{1}{2}\|y - y^*\|^2 \end{aligned}$$

By Lemma 1, we can obtain the following inequality

$$V(x,y) \geq \frac{1}{2}\|x-x^*\|^2 + \frac{1}{2}\|y-y^*\|^2. \tag{8}$$

Calculating the derivative of V along the solution of (7),

$$\begin{aligned}
\tfrac{dV}{dt} &= \tfrac{\partial V}{\partial x}\tfrac{dx}{dt} + \tfrac{\partial V}{\partial y}\tfrac{dy}{dt}\\
&= -2[\nabla f(x) - g'(x)^T(y-g(x))^+ - \nabla f(x^*) + g'(x^*)^T y^* + x - x^*]^T\\
&\quad \{(I-P)[\nabla f(x) - g'(x)^T(y-g(x))^+] - Px + Q\}\\
&\quad -[y-(y-g(x))^+]^T[(y-g(x))^+ - 2y^* + y]\\
&= -[\nabla f(x) - \nabla f(x^*) + x - x^* + g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]^T\\
&\quad \{(I-P)[\nabla f(x) - \nabla f(x^*)] + (I-P)[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad +P(x-x^*)\}\\
&\quad -[y-(y-g(x))^+]^T[y-(y-g(x))^+ + 2(y-g(x))^+ - 2y^*]\\
&= -2\Big\{[\nabla f(x) - \nabla f(x^*)]^T(I-P)[\nabla f(x) - \nabla f(x^*)]\\
&\quad +[\nabla f(x) - \nabla f(x^*)]^T(I-P)[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad +[\nabla f(x) - \nabla f(x^*)]^T P(x-x^*)\\
&\quad +(x-x^*)^T(I-P)[\nabla f(x) - \nabla f(x^*)]\\
&\quad +(x-x^*)^T(I-P)[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad +(x-x^*)^T P(x-x^*)\\
&\quad +[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]^T(I-P)[\nabla f(x) - \nabla f(x^*)]\\
&\quad +[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]^T(I-P)[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad +[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]^T P(x-x^*)\Big\}\\
&\quad -\parallel y-(y-g(x))^+ \parallel^2 - 2[y-(y-g(x))^+]^T[(y-g(x))^+ - y^*]
\end{aligned}$$

Noting that $(I-P)^2 = I-P$, $P^2 = P$ and $P(I-P) = 0$, we have

$$\begin{aligned}
\tfrac{dV}{dt} &= -2\Big\{[\nabla f(x) - \nabla f(x^*)]^T(I-P)^2[\nabla f(x) - \nabla f(x^*)]\\
&\quad +2[\nabla f(x) - \nabla f(x^*)]^T(I-P)^2[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad +[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]^T(I-P)^2[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad +(x-x^*)^T P^2(x-x^*)\Big\} - 2(x-x^*)^T[\nabla f(x) - \nabla f(x^*)]\\
&\quad -2(x-x^*)^T[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad -\parallel y-(y-g(x))^+ \parallel^2 - 2[g(x)-(g(x)-y)^+]^T[(y-g(x))^+ - y^*]\\
&= -2\parallel (I-P)[\nabla f(x) - \nabla f(x^*)] + (I-P)[g'(x^*)^T y^* - g'(x)^T(y-g(x))^+]\\
&\quad +P(x-x^*)\parallel^2 - 2(x-x^*)^T[\nabla f(x) - \nabla f(x^*)] - 2(x-x^*)^T g'(x^*)^T y^*\\
&\quad -2(x-x^*)^T g'(x)^T(y-g(x))^+\\
&\quad -\parallel y-(y-g(x))^+ \parallel^2 - 2g(x)^T(y-g(x))^+ + 2g(x)^T y^*\\
&\quad -2[(g(x)-y))^+]^T(y-g(x))^+ - 2[(y-g(x))^+]^T y^*\\
&= -2\parallel (I-P)[\nabla f(x) - g'(x)^T(y-g(x))^+] - Px + Q\parallel^2\\
&\quad -\parallel y-(y-g(x))^+ \parallel^2 - 2(x-x^*)^T[\nabla f(x) - \nabla f(x^*)] - [(y-g(x))^+]^T y^*\\
&\quad -2(x-x^*)^T g'(x^*)^T y^* - 2(x-x^*)^T g'(x)^T(y-g(x))^+ - 2g(x)^T(y-g(x))^+\\
&\quad +g(x)^T y^*\\
&= -2\parallel (I-P)[\nabla f(x) - g'(x)^T(y-g(x))^+] - Px + Q\parallel^2\\
&\quad -\parallel y-(y-g(x))^+ \parallel^2 - 2(x-x^*)^T[\nabla f(x) - \nabla f(x^*)] - [(y-g(x))^+]^T y^*\\
&\quad +2[g(x) - g(x^*) - g'(x^*)(x-x^*)]^T y^* + g(x^*)^T y^*\\
&\quad +2[g(x^*) - g(x) - g'(x)(x^*-x)]^T(y-g(x))^+ - 2g(x^*)^T(y-g(x))^+
\end{aligned}$$

Since $f(x)$ and $-g_i(x)$ are convex function, we have

$$\begin{array}{l} (x-x^*)^T[\nabla f(x)-\nabla f(x^*)] \geq 0 \\ g(x^*)-g(x)-g\prime(x)(x^*-x) \leq 0 \\ g(x)-g(x^*)-g\prime(x^*)(x-x^*) \leq 0 \end{array} \qquad (9)$$

In addition, it is easy to verify $[(g(x)-y)^+]^T(y-g(x))^+ = 0, \quad g(x^*)^T y^* = 0,$ and $[(g(x)-y)^+]^T y^* \geq 0$, $g(x^*)^T[y-(g(x))^+]^T \geq 0$. Thus,

$$\begin{array}{l} \frac{dV}{dt} \leq -2 \parallel (I-P)[\nabla f(x)-g'(x)^T(y-g(x))^+] - Px + Q \parallel^2 \\ \quad - \parallel y-(y-g(x))^+ \parallel^2 < 0, \qquad \forall\ (x,y) \neq (x^*,y^*). \end{array} \qquad (10)$$

Thus, the neural network is asymptotically stable for any initial points. This completes the proof.

4 Numerical Examples

In this section, we will give two examples to demonstrate the effectiveness of the proposed neural network.

Example 1. Consider the following nonlinear programming problem.

$$\begin{array}{ll} \text{minimize} & f(x) = x_1^4 + (x_2-2)^2 + (x_2-x_3)^2 \\ \text{subject to} & 3x_1 - x_2 + 4x_3 = 6 \\ & -x_1 + 2x_2 + 3x_3 = 10 \\ & g_1(x) = x_1^2 + x_2^2 + x_3^2 - 8 \leq 0 \\ & g_2(x) = 2x_1^2 + 2x_2^2 + x_3^2 - 12 \leq 0 \end{array}$$

This problem has an optimal solution $x^* = (0,\ 2,\ 2)^T$. The simulation result shows the neural network (7) converges to the optimal solution. The Fig. 1 is the state trajectory of neural network (7).

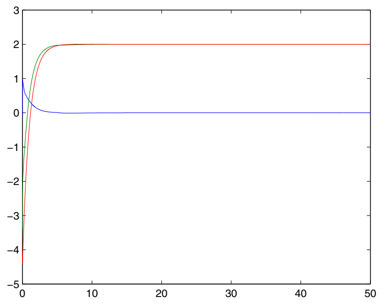


Fig. 1. Transient behavior of (7) in Example 1

Example 2. Consider the following nonlinear programming problem.

$$\begin{array}{ll} \text{minimize} & f(x) = (x_1-1)^4 + (x_1-2x_2)^4 + (4x_2-x_3)^4 + (x_4-x_5)^4 \\ \text{subject to} & 2x_3 - x_4 + x_5 = 4 \\ & -x_1 + 2x_2 + 2x_3 + x_4 + x_5 = 4 \\ & g_1(x) = x_1^2 + x_3^2 + x_4^2 - 5 \leq 0 \\ & g_2(x) = x_1^2 + 4x_2^2 + 4x_5^2 - 3 \leq 0 \end{array}$$

This problem has an optimal solution $x^* = (1,\ 0.5,\ 2,\ 0,\ 0)^T$. The simulation result shows the neural network (7) converges to the optimal solution. The Fig. 2 is the state trajectory of neural network (7).

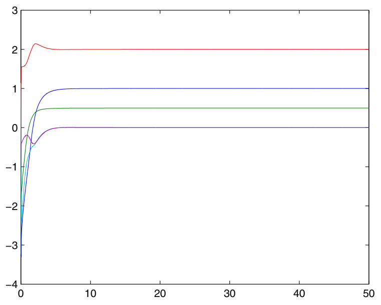


Fig. 2. Transient behavior of (7) in Example 2

5 Conclusion

In this paper, we present a new neural network for solving nonlinear convex programming problems. The proposed neural network is asymptotically stable in sense of Lyapunov. Comparing with other neural networks for nonlinear convex optimization, the proposed neural network has fewer neurons and simpler architecture. Some examples are given to illustrate the effectiveness of the proposed neural network.

Acknowledgement

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.

References

1. Bazaraa, M., Sherali, H., Shetty, C.: Nonlinear programming: Theory and algorithms. Wiley, New York (1993)
2. Hopfield, J., Tank, D.: Neural computation of decisions in optimization problem. Biol. Cybern. 52, 141–152 (1985)

3. Cichocki, A., Unbehauen, R.: Neural networks for optimization and signal processing. Wiley, New York (1993)
4. Tank, D., Hopfield, J.: Simple neural optimization networks: an A/D converter, signal decision circuit. IEEE Trans. Cir. Sys. 33(5), 533–541 (1986)
5. Kennedy, M., Chua, L.: Neural network for nonlinear programming. IEEE Trans. Cir. Sys. 35(5), 554–562 (1988)
6. Zhang, S., Constantinides, A.: Lagrange programming neural networks. IEEE Trans. Cir. Sys. 39(7), 441–452 (1992)
7. Xia, Y.: A new neural network for solving linear and quadratic programming problems. IEEE Trans. Neural. Net. 7(6), 1544–1547 (1996)
8. Liu, S., Wang, J.: A simplified dual neural network for quadratic programming with its KWTA application. IEEE Trans. Neu. Net. 17(6), 1500–1510 (2006)
9. Leung, Y., Gao, X.: A high-performance feedback neural network for solving convex nonlinear programming problems. IEEE Trans. Neu. Net. 14, 1469–1477 (2003)
10. Yang, Y., Cao, J.: A feedback neural network for solving convex constraint optimization problems. App. Mathe. Compu. 201(1-2), 340–350 (2008)
11. Liu, Q., Wang, J.: A one-layer recurrent neural network with a discontinuous activation function for linear programming. Neur. Compu. 20(5), 1366–1383 (2008)
12. Liu, Q., Wang, J.: A one-layer recurrent neural network with a discontinuous hardlimiting activation function for quadratic programming. IEEE Trans. Neu. Net. 19(4), 558–570 (2008)
13. Tao, Q., Cao, J., Xue, M., Qiao, H.: A high performance neural network for solving nonlinear programming problems with hybrid constraints. Phys. Let. A 288, 88–94 (2001)
14. Effati, S., Baymani, M.: A new nonlinear neural network for solving quadratic programming problems. Appl. Mathe. Compu. 165, 719–729 (2005)
15. Xia, Y., Leung, H., Wang, J.: A projection neural network and its application to constrained optimization problems. IEEE Trans. Cir. Sys. 49(4), 447–458 (2002)
16. Xia, Y., Wang, J.: A recurrent neural network for nonlinear convex optimization subject to nonlinear inequality constraints. IEEE Trans. Cir. Sys. 51(7), 1385–1394 (2004)
17. Xia, Y., Wang, J.: A recurrent neural network for nonlinear convex optimization subject to linear constraints. IEEE Trans. Cir. Sys. 16(2), 379–386 (2005)
18. Liang, X.: A recurrent neural networks for nonlinear continuously differentiable optimization over a compact convex subset. IEEE Trans. Neur. Net. 12, 1487–1490 (2001)
19. Gao, X.: A novel neural network for nonlinear convex programming. IEEE Trans. Neu. Net. 15, 613–621 (2004)
20. Hu, X., Wang, J.: Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network. IEEE Trans. Neu. Net. 17(6), 1487–1499 (2006)
21. Hu, X.: Applications of the general projection neural network in solving extended linear-quadratic programming problems with linear constraints. Neurocom. 72, 1131–1137 (2009)
22. Avriel, M.: Nonlinear programming: Analysis and Methods. Prentice-Hall, Englewood Cliffs (1976)
23. Miller, R., Michel, A.: Ordinary differential equations. Academic, New York (1982)

The AIC Based on the Neural Network Filter

Yunfeng He[1], Gaoshun Song[1], Changming Wang[1], and Junsheng Jiao[2]

[1] Department of Precise Instrument, Nanjing University of Sci. & Tech., Nanjing, 210094, China
[2] Hangzhou Application Acoustics Graduate School, Hangzhou, 310012, China

Abstract. It is impossible to recognize the direction of the target accurately when strong directional interference appears underwater. The adaptive interference cancellation based on the neural network filter is used to process the signal to solve the problem. In order to improve the calculation precision of the network, stochastic gradient BP algorithms is used to adjust the weights and threshold values. Finally, the simulation and the corresponding conclusion are presented by computer.

Keywords: neural network; adaptive interference cancellation filter; BP; simulation.

1 Introduction

When the noise under water is nondirectional or very weak, use the current conventional algorithm can distinguish the target and estimated the direction angle very well. But when there is a strong directivity interference, Array Signal Processing can not completely suppress this interference, and the 'false target' spectral peak appeared on the energy spectrum. It will produce blind spots in the detecting space and make flank arrays can not estimate the direction of the target underwater. In order to reduce the spectral peak of the interference on the energy spectrum space, and relatively improve the spectral peak of the target underwater, the adaptive interference counteract should be used to proceeding the output signal of beam or elements.

As in the actual environment, the interference in main channel and reference channel is a non-stationary, non-Gaussian distribution and it often presents nonlinear approaching, nonlinear and adaptive filter must be use in the interference cancellation [1], and the neural network filter is worth studying. Neural network filter can be a multi-layer perceptron (MLP), and also can be a radial basis function (RBF) networks [2], the adaptive interference cancellation (AIC) based on the MLP is discussed in this paper.

2 AIC Based on the MLP

The input and output functions of the hidden layer element of the MLP can be chosen as linear function, breadth-limit function and sigmoid function. Three layers single-output will be discussed, and the Sigmoid function is taken as input and output functions of the

© Springer-Verlag Berlin Heidelberg 2011

hidden layer element, the linear function is taken as input and output functions of the output layer element. If replace the FIR of conventional AIC with the MLP, we will get the AIC based on the MLP, the structure is shown in Figure 1. From figure 1, there is one input layer, one hidden layer and one output layer in MLP. Assume there are N elements in the input, and M elements in hidden layer.

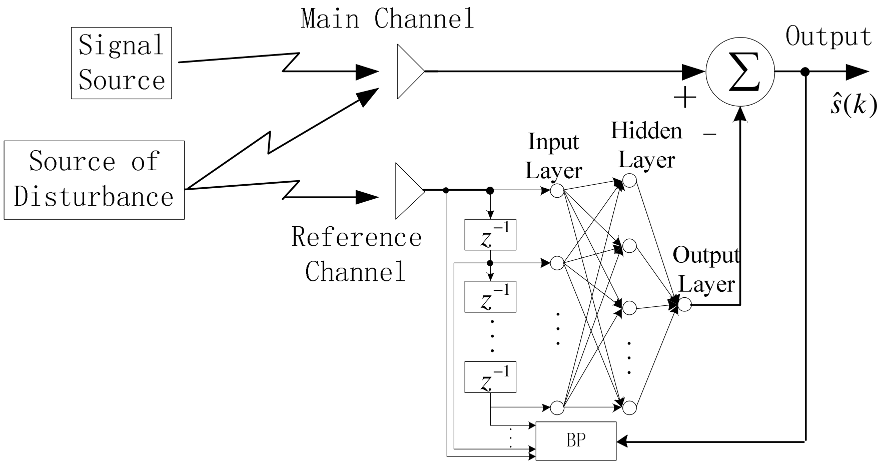


Fig. 1. AIC based on the MLP

After N-1 time delay, get the MLP input vector from the reference input as:

$$X(k) = [x(k) \quad x(k+1) \quad \cdots \quad x(k+N\text{-}1)]^T \tag{1}$$

Take the input layer neurons as a linear function, at time *k*, the output of MLP neuron in each input layers is:

$$\boldsymbol{O}_0(k)=[x(k) \quad x(k+1) \quad \cdots \quad x(k+N-1)]^T \tag{2}$$

Take the weight vectors of the first *j* a element in hidden layer as:
$\boldsymbol{W}_j(k)=[w_{1,j}(k) \quad w_{1,j}(k) \quad \cdots \quad w_{N,j}(k)]^T$, $(1\le j\le M)$ the element's net input is:

$$I_j(k)=\boldsymbol{W}_j(k)\boldsymbol{O}_0(k)+\theta_j(k) \tag{3}$$

Where θ_j is the first *j* a element threshold. Due to the hidden layer uses the sigmoid function, first *j* a element output of the hidden layer is:

$$o_j(k)=\frac{1}{1+\exp[-I_j(k)]} \tag{4}$$

Write the output of each element in the hidden layer in the vector, we get:

$$\boldsymbol{O}_1(k)=[o_1(k) \quad o_2(k) \quad \cdots \quad o_M(k)]^T \tag{5}$$

Suppose that the weight vectors of element in output layer are:
$\boldsymbol{V}(k)=[v_1(k) \quad v_2(k) \quad \cdots \quad v_M(k)]^T$, the element's output is:

$$y(k)=\boldsymbol{V}^T(k)\boldsymbol{O}_1(k)+\gamma(k) \tag{6}$$

Where γ is the element threshold of the output layer.

If the input of neural network and the element's weights and thresholds are given, can calculate the output of the MLP by using the formula above, which is called forward computation of the network, for the input information of the network is started by the input layer, then pass on to the next layer by layer, and there is no information feedback process. Defined *k* a error function of the AIC sample is:

$$e(k)=\frac{1}{2}[d(k)-y(k)]^2 \tag{7}$$

Where $d(k)$ is the *k* a input data of main channel.

Under normal circumstances, the initial weights and thresholds of the neural network are random, so, if get network output $y(k)$ in this way, the error function value will be larger, in another word, network calculation precision is very poor. So after determining the structure of the network, we could only adjust the weights and the thresholds of the element to reduce output deviation $(d(k)-y(k))$ gradually. In order to enhance the precision of network, stochastic gradient BP algorithms are used to adjust the weights and threshold values.

3 Stochastic Gradient BP Algorithm

With stochastic gradient BP algorithm, the beat should be adjusted at the *k-th* weight by extracting a group of training data which are marked with $X(k)$ from the input sequences. Then from the input sequences, a new group of training data called $X(k+1)$ is extracted, the following *(k+1)*-th weight could be adjusted. However it is different from the offline batch whose process of learning is separated from the scene, and only all the training data are called, each weight could adjust its beat. With stochastic gradient BP algorithm, the weights and threshold values are adjusted along the negative gradient direction of error function, as:

$$\Delta \boldsymbol{W}_j(k)=-\eta\frac{\partial e(k)}{\partial \boldsymbol{W}_j(k)}\ ;\ \Delta \boldsymbol{\theta}(k)=-\eta\frac{\partial e(k)}{\partial \boldsymbol{\theta}(k)} \tag{8}$$

$$\Delta \boldsymbol{V}(k)=-\eta\frac{\partial e(k)}{\partial \boldsymbol{V}(k)}\ ;\ \Delta \gamma(k)=-\eta\frac{\partial e(k)}{\partial \gamma(k)} \tag{9}$$

Where η is the learning rate, and $0<\eta<1$. From the expression (**7**), the weights and threshold values adjustment amount of the output neurons could be calculated firstly, then calculate the weights and threshold values of the neurons in the hidden layer. And the calculation process is reversed. Will get:

$$\Delta \boldsymbol{V}(k)=\eta[d(k)-y(k)]\frac{\partial y(k)}{\boldsymbol{V}(k)}=\eta[d(k)-y(k)]\boldsymbol{O}_1(k) \tag{10}$$

$$\Delta\gamma(k)=\eta[d(k)-y(k)]\frac{\partial y(k)}{\gamma(k)}=\eta[d(k)-y(k)] \tag{11}$$

$$\begin{aligned}\Delta\theta_j(k) &= \eta[d(k)-y(k)]\frac{\partial e(k)}{\partial\theta_j(k)} = \eta[d(k)-y(k)]v_j(k)\frac{\partial o_j(k)}{\theta_j(k)} \\ &= \eta[d(k)-y(k)]v_j(k)I_j(k)[1-I_j(k)]\end{aligned} \tag{12}$$

$$\begin{aligned}\Delta \boldsymbol{W}_j(k) &= \eta[d(k)-y(k)]\frac{\partial y(k)}{\boldsymbol{W}_j(k)} = \eta[d(k)-y(k)]v_j(k)I_j(k)[1-I_j(k)]\boldsymbol{O}_0(k) \\ &= \eta[d(k)-y(k)]v_j(k)I_j(k)[1-I_j(k)]\boldsymbol{X}(k) \quad (j=1,2,\cdots M)\end{aligned} \tag{13}$$

From all the formulas above, after each adjustment values are calculated, we could calculate the weights on the latest adjusted beat by iterative method. For example, the weight value of neuron in the hidden layer is:

$$\boldsymbol{W}_j(k+1)=\boldsymbol{W}_j(k)+\Delta\boldsymbol{W}_j(k) \tag{14}$$

Error function $e(k)$ usually has multiple local minimum points, and sometimes platform. The steepest descent algorithm based on BP algorithm may make the neuron weights and thresholds converge to a local minimum point, or stay in one platform. Therefore, inertial item should be added to the BP algorithm to prevent them. With inertial item, weights' updated formula of hidden layer neurons is:

$$\boldsymbol{W}_j(k+1)=\boldsymbol{W}_j(k)+\Delta\boldsymbol{W}_j(k)+\alpha\Delta\boldsymbol{W}_j(k-1) \tag{15}$$

Where $0<\alpha<1$, which is called inertial coefficient. Hidden neurons thresholds value iteration formula and the iterative formula of the output layer neurons weights and thresholds can be modeled on the style as:

$$\theta_j(k+1)=\theta_j(k)+\Delta\theta_j(k)+\alpha\Delta\theta_j(k-1) \tag{16}$$

$$\boldsymbol{V}_j(k+1)=\boldsymbol{V}_j(k)+\Delta\boldsymbol{V}_j(k)+\alpha\Delta\boldsymbol{V}_j(k-1) \tag{17}$$

$$\gamma(k+1)=\gamma(k)+\Delta\gamma(k)+\alpha\Delta\gamma(k-1) \tag{18}$$

Based on stochastic gradient BP algorithm, we can get the output from MLP only by a set of training data under any adjusted beat. Then error component of the beat could be used to construct error function. Another on-line BP algorithm is that p-groups of real-time training data could be picked so as to get p outputs of MLP. We can get p error components, and its error function is:

$$e(k)=\frac{1}{2P}\sum_{p=1}^{P}[d_p(k)-y_p(k)]^2 \tag{19}$$

The more groups of training data set, the wider will the average scope be. On the contrary, the adaptability of the network will be strengthened as the average scope is smaller.

4 Computer Simulation

Supposed that the useful signals of AIC main output are 400Hz narrow band ones. Interference in the reference input frequency a narrow band signal which is equal to 240Hz. Through nonlinear system, the interference can get nonlinear interference from AIC main channel, the main channel and reference channel input signal are:

$$x(t)=\sin(2\pi\cdot 400t)+\sin^2(2\pi\cdot 240t)+\cos(2\pi\cdot 240t+0.5)$$
$$v(t)=\sin(2\pi\cdot 240t+1)$$

After the non-linear system makes a difference on the interference of reference channel, the interference of main output has components when its frequency is 0Hz, 240Hz and 480Hz. With AIC which is based stochastic gradient BP algorithm, the linear and nonlinear interferences of the main channel and the reference could offset. The AIC based on NLMS algorithm can only cancel linear interference component which has 240 Hz frequency and comes from main output, as figure 2 shows:

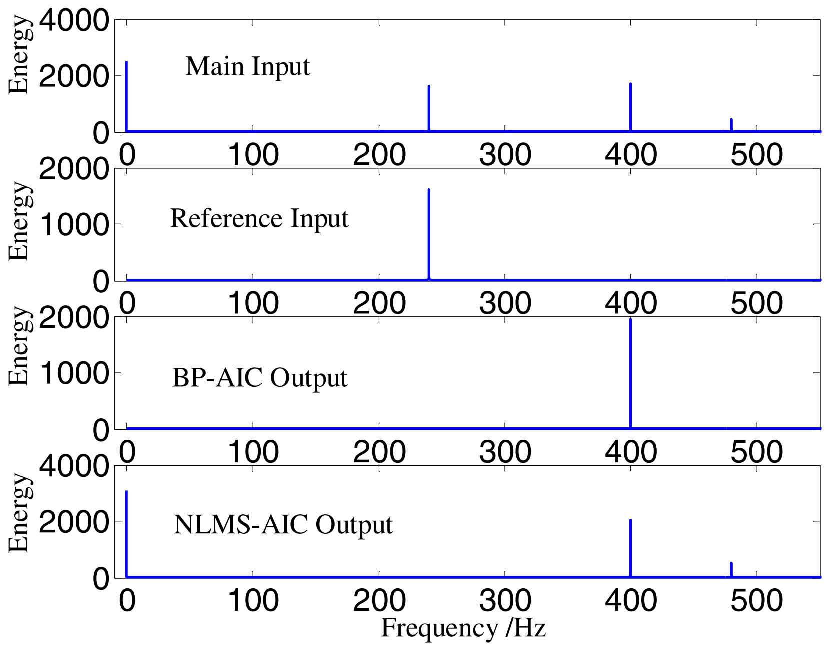


Fig. 2. Input and output power spectrum of two kinds of AIC

5 Conclusion

Compared with conventional linear algorithm, AIC is much better by using a neural network nonlinear algorithm in nonlinear interference offset. However, for broadband interference offset, the convergence speed of nonlinear algorithm is too slow. And its interference ability is poor than conventional one. Therefore, the nonlinear AIC has not yet been applied to broadband acoustic signal processing so far.

References

1. Chu, H.S., An, C.K.: Design of the adaptive noise canceler using neural network with backpropagation algorithm science and technology. In: Proceedings of the Third Russian-Korean International Symposium on KORUS 1999, June 22-25, vol. 2, pp. 762–764 (1999)
2. Liang, T., Guoxiang, D., Delong, Z.: Neural network for adaptive noise cancellation-related. Microelectronics and Computer 5, 20–23 (1995)
3. Xingjun, Y., Junli, Z.: Artificial Neural Networks and blind signal processing. Tsinghua University Press, Beijing (2003)
4. Lou, G., HuiYong, L., ZhengWu, X.: Applications of Steepest Descent Arithmetic in Designing the MTD filters. Journal of University of Electronic Science and Technology of China, 593–595 (2005)
5. Changhong, D.: The Neural Network by Matlab and the application. National Defence Industry Press, Beijing (2005)
6. Wang, J.H., Yang, G.: A new magnetic compass calibration algorithm using neural networks. Measurement Science and Technology 17(1), 153–160 (2006)
7. Vaccaro, R.J. (ed.): The past, present, and future of underwater acoustic signal processing. IEEE Signal Processing Magazine 15(4), 21–51 (1998)
8. Samarasinghe, S.: Neural networks for applied sciences and engineering: from fundamentals to complex pattern recognition. Machine Industry Press, Beijing (2010)
9. Kumar, S.: Neural Networks. Tsinghua University Press, Beijing (2006)
10. Zhongzhi, S.: Neural Networks. Higher Education Press, Beijing (2009)
11. Simon, H.: Neural Networks and Machine Learning. Machine Industry Press, Beijing (2009)
12. Zongquan, Y., Han, Y.: Neural Networks Control. Xidian University Press, Xi'an (2009)

Ranked Neuro Fuzzy Inference System (RNFIS) for Information Retrieval

Asif Nawaz and Aasia Khanum

College of Electrical and Mechanical Engineering
National University of Sciences and Technology (NUST)
Islamabad, Pakistan
aasifnawaz@gmail.com, aasia@ceme.nust.edu.pk

Abstract. The paper presents a novel approach to informational retrieval based on a synergy of knowledge-based models, set theoretic models, and vector space models of domain within a Fuzzy Logic framework. An input query is expanded to multiple synonym queries based on query semantics. Each document in the collection is divided into different zones with different relative importance assigned to each zone indicating its role in the query. Fuzzy rule bases are applied to each zone with parameters derived from vector space models and semantic query expansion. Fuzzy inference procedure outputs the relevance rank of each zone in satisfying the query. The relevance ranks of different zones are aggregated using the Ordered Weighted Averaging (OWA) operator to get the overall relevance rank of the complete document. The documents are ranked according to their relevance. The system has been tested on a standard dataset and has been demonstrated to show improved performance over typical vector space based approaches.

Keywords: Information Retrieval, Fuzzy Logic, Natural Language Processing.

1 Introduction

Information Retrieval (IR) refers to search for information in large collection of unstructured or semi-structured documents according to user's requirements [1]. IR is a more complex problem than simple database retrieval. A database (e.g. relational database) imposes a well-defined, semantically un-ambiguous structure on its contents, making it amenable for computer processing. In case of IR there is no hard constraint on the structure of the document or its content [2]. An IR system aims to maximize the number of relevant documents retrieved while serving a user query, simultaneously minimizing the number of irrelevant documents [1][2].

Complexity of documents can be reduced by text operations to obtain a logical view of document [5]. The most complete logical view of a document is full text called bag of words representation but its usage usually implies higher cost. In a large document collection, compact logical view of documents can be achieved by means of transformations like eliminating stop words, simplifying noun groups, and stemming. The keywords or the index terms usually represent the documents that may be specified by a human or can be extracted from the text [6][1].

© Springer-Verlag Berlin Heidelberg 2011

There are four classic models of document and query representation in IR [1] [2] [14]: Boolean Model, Vector Model, Probabilistic Model, and Knowledge-based (or linguistic) Model. The Boolean models compose a query by combining one or more distinct query terms with the help of operators like AND and OR. The model provides clean formalism and simplicity. The problem with Boolean search is that using AND operators produces high precision but low recall searches, while using OR operators tends to produce low precision but high recall searches and it is hard or quiet impossible to have a tradeoff between the two [2]. Another problem with Boolean model is that it only records the presence or absence of query term in a document while actually it is desirable to give more weights to documents that have higher frequency of query terms [6]. To be able do this we have to keep the information of term frequency. Boolean queries only return the relevant document set but we desire to have an effective method to rank the documents in the set with respect to relevance with the user query [13].

In vector space model [15] a single vector represents each terms of document or query in t-dimensional space, total terms are represented by t. The frequency of term in document helps to determine its score. This measure is referred to as term frequency denoted by $tf_{t,d}$. Raw term frequency experiences a major issue: all terms will be considered equally important when calculating the score of document on query q. To overcome this issue, the weight of the term having higher collection frequency are considered to be less important and its weight need to be scaled down, total documents containing the term is represented by df. Let N is total documents in collection, inverse document frequency (idf) of term t is [2]

$$idf_t = \log \frac{N}{df_t} \tag{1}$$

In order to produce the net weight of term t, a combination of tf and idf is used [2]

$$tf\_idf_{t,d} = tf_{t,d} \times idf_t \tag{2}$$

To account for the documents of different lengths tf and idf are generally used in their normalized form. The net score of document over all query terms is obtained by summing up tf_idf of all the terms. [2] i.e.

$$Score\ (q,d) = \sum_{t \epsilon Q} tf\_idf_{t,d} \tag{3}$$

Similarity in user query and document is usually calculated by means of the cosine formula. As the method is not constrained to binary weights, so it is possible to get ranked similarity. However, there is no notion of relative ordering of terms. e.g. Car is on the Road and Road is on the Car are same in each vector representation [2]. Moreover, the model requires an exact match between terms and does not consider semantic similarity.

The probabilistic model focuses on the conditional probability of document relevance to user query. However, the accurate estimation of initial probabilities is a major obstacle in successful implementation of these models. Knowledge-based models make use of natural language knowledge to enhance retrieval results, e.g. syntax knowledge, synonym knowledge etc. While these models hold great promise, enclosing them in a formal framework is a formidable challenge.

Although most of the mainstream commercial systems employ the above described models for IR, there is a need to improve the performance of these systems by using improved modeling techniques. In this regard, the field of Soft Computing holds great promise to exploit the peculiar characteristics of IR domain. Traditional techniques do not adequately address the inherent imprecision and vagueness in document representation, query formulation, and document-query relevance. Fuzzy Logic is a Soft Computing paradigm that provides adequate constructs for reasoning with a tolerance for imprecision and vagueness.

Some approaches have been proposed in research literature to handle IR problem with the help of Fuzzy Logic. For example, [14] utilizes the concept of fuzzy proximity to incorporate flexible definition of relevance between document and query terms. [15] addresses the relevance issue by means of fuzzy division of relations. In [12] numeric query weights have been replaced by fuzzy linguistic weights to denote relative importance of query terms. [13] combines fuzzy relations and ontologies for IR. Our proposed approach (RNFIS) is most similar to [3] which uses fuzzy version of vector space model; but our approach introduces a combination of vector space and knowledge-based parameters which are applied to semantic document zones unlike [3] which applies the parameters to the entire document together.

In this paper fuzzy rule based approach Ranked Neuro Fuzzy Inference System (RNFIS) is presented and a list of ranked documents are returned with respect to query relevance. The model is hybrid, using vector space for information retrieval and fuzzy enhanced Boolean theory for document scoring. The fuzzy inference system combines vector space parameters with semantic parameters calculated from semi-structured documents for exploiting the inherent imprecision of the domain. The proposed model gives simplicity of logic based models and the performance and flexibility of vector space models.

Following sections are organized as; proposed system is presented in section 2, experimental setup and results are described in section 3. Finally conclusion of paper is given in section 4.

2 Proposed Scheme

Ranked Neuro Fuzzy Inference System (RNFIS), shown in Fig-1. The proposed scheme is Fuzzy Logic based that inherits from Boolean logic. It allows imprecise matching of documents with query. IR is modeled by defining a Fuzzy Set against each query and every document will have a degree of membership in that set [10]. Other features included in this model for better relevance scoring of documents are given below.

2.1 Weighted Zone Scoring

Semi structured document is represented through different markup tags that demarcate zones of document. Different weights are assigned to the zones according to their role in the application domain. For example, in case of research publications the Abstract section (Zone) has higher weight than the Reference section (Zone).

2.2 Semantic Matching

Vector Space Models based on exact matching of query terms cannot retrieve all documents relevant to user query because of different words used in the documents to address the same idea. The proposed approach exploits the notion of synonym queries where the original query term is replaced by its synonym to generate multiple queries.

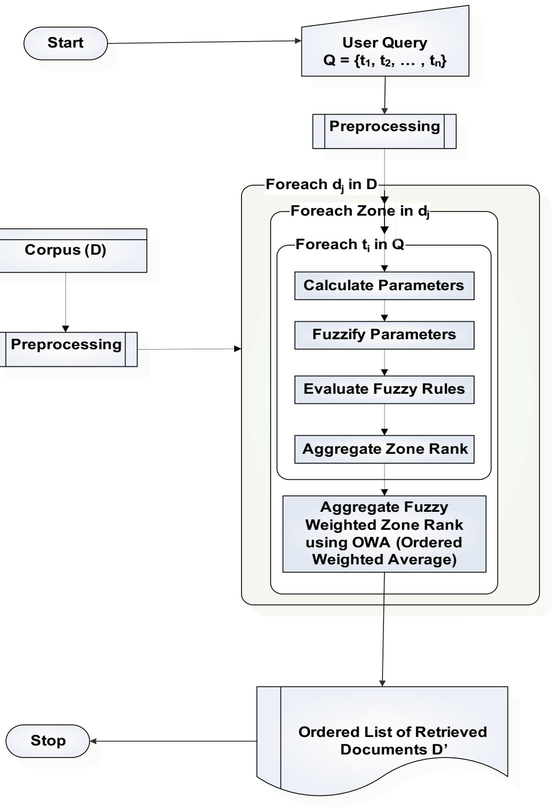


Fig. 1. Flowchart of RNFIS

2.3 Fuzzy Linguistic Variables

Various inputs and outputs are represented by fuzzy linguistic variables [13]. The input variables are (term frequency) tf, (inverse document frequency) idf, overlap (number of query terms found in document), and Zone. All of these variables are represented by Gaussian membership functions. Fig-2 shows an example.

2.4 Fuzzy Inference System

There are two types of fuzzy inference systems supported, Mamdani's and Sugeno's. Mamdani's framework provides simplicity but Sugeno's inference system gives more

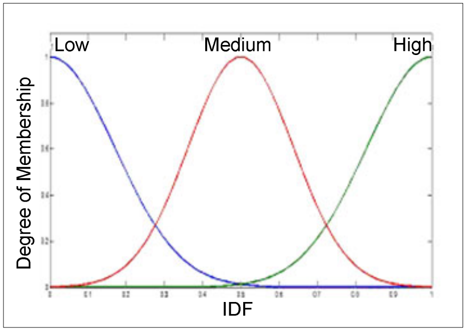


Fig. 2. Representation of Parameters

optimal results so we have used Sugeno's method. The proposed Sugeno-type systems was created, trained and test by using ANFIS (Adaptive Neuro Fuzzy Inference System). Its rules provide mapping of input variables to output variables[6]. The form of rule in Sugeno's fuzzy model is.

If x and y are two input variables, the Output will be

$$z = ax + by + c \tag{4}$$

After all rules have been evaluated, their results are combined by the aggregation operation to obtain the final relevance rank of each document. The aggregation operation again produces a fuzzy set over values range. This is defuzzified to determine a crisp value from using the weighted average operation.

3 Experimental Results and Evaluation

3.1 Document Corpus

A gold standard Cystic Fibrosis Corpus (CFC)[5] is used to evaluate the proposed system. The CFC database consists of six files: cf74 to cf79 containing 1239 documents. There are 100 queries with their set of relevant documents as answered by 4 different domain experts at scale of 0, 1, 2 from irrelevant, marginally relevant, and relevant.

The CFC database is in SGML format whose parsers is not available for Java. As the proposed system is developed in Java, so a manual parser is written to convert these documents from existing format into XML format, whose parsers are available for almost all languages.

3.2 FIS Training

The data used for training the ANFIS is illustrated in Table 1. Here the score has been calculated as

$$\text{Score} = \text{tf} \times \text{idf} \times \text{overlap} \times \text{zone} \tag{5}$$

The Neuro FIS is trained based on different parameters. Hybrid optimization, which is a mixture of least square and back propagation gradient descent method, is used to train the system. Other parameters used to train the FIS are error threshold and required no of training epochs.

We have used Grid Partition method for generating RNFIS. Number of input membership function for each input variable is chosen that depend upon the desired performance level of the system. We have used Gaussian function for input variables and linear membership function type is chosen for output variables.

Table 1. Training Data

tf	idf	zone	overlap	Score
0.00558659	0.32411947	0.5	1.096710205	0.000992919
0.00588235	0.32411947	1	1.096710205	0.002090971
0.00900900	0.32411947	1	1.096710205	0.003202389
0.00740741	0.432460612	0.5	1.49271137	0.002390885
0.00740741	0.432460612	4	1.248975876	0.016003937
0.00740741	0.432460612	16	1.49271137	0.033086831
0.00769231	0.432460612	2	1.587962963	0.010565099

3.3 Evaluation Parameters

To assess the effectiveness of proposed system, we use the following statistics about the system's returned result for a query:

Precision:
This is the ratio of the returned results that are relevant to the information need to the completed fetched results [1,11].

$$Precision = \frac{Number\ of\ Relevant\ Items\ Retrieved}{Total\ Retrieved\ Items} \tag{6}$$

Recall:
This is the ratio of the relevant documents in the collection that were returned by the system to the completed fetched results [1, 11].

$$Recall = \frac{Number\ of\ Relevant\ Items\ Retrieved}{Total\ Relevant\ Items} \tag{7}$$

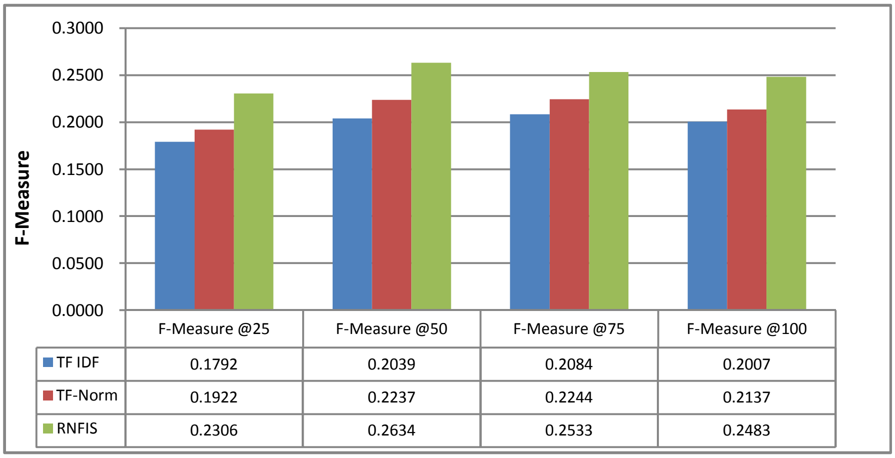
F-measure:
The weighted harmonic mean that trades-off between precision P and recall R [2].

$$F = \frac{(\beta^2)PR}{\beta^2(P+R)} \tag{8}$$

Where $\beta^2 = \frac{1-\alpha}{\alpha}$, α ε [0,1] and thus $\beta^2 \in [0, \infty]$. It is usually used with β=1 represented by $F_{\beta=1}$ [2].

$$F_{\beta=1} = \frac{2PR}{P+R} \tag{9}$$

Fig-3 shows the F-measure of proposed system with traditional vector space approaches using tf-idf and tf-normalized. A higher value of F-measure for the proposed system means that the balance between precision and recall is compared to traditional methods.



	F-Measure @25	F-Measure @50	F-Measure @75	F-Measure @100
🟦 TF IDF	0.1792	0.2039	0.2084	0.2007
🟥 TF-Norm	0.1922	0.2237	0.2244	0.2137
🟩 RNFIS	0.2306	0.2634	0.2533	0.2483

Fig. 3. Comparison of F-Measure @ 25, 50, 75 & 100 queries

The Mean Average Precision (MAP):

For a single information need, the precisions of top k documents is averaged by total relevant documents retrieved till that point is said to be Mean Average Precision (MAP). For instance, From the set of relevant documents $(d_1, d_2, d_3, \ldots, d_n)$ for user information need $q_j \in Q$ and from the top most results R_{jk} is a set of ranked retrieval results until the document R_k is retrieved [2].

$$\text{MAP (Q)} = \frac{1}{|Q|} \sum_{J=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} \text{Precision}(R_{jk}) \tag{10}$$

Precision and Mean Average Precision (MAP) of our proposed approach against all the queries is compared with two existing approaches in Table 2. It is clear that the proposed approach achieves better precision than other approaches.

The comparison is illustrated graphically in Fig 4 which shows that the peaks of precision values of our proposed approach are greater than those of other approaches at most of the places.

Figure 5 shows the MAP of 25, 50, 75 and 100 queries showing that RNFIS has better performance against the exiting approaches.

Table 2. Mean Average precision of 100 queries in brief using all approaches

Query #	TF IDF	TF IDF Normalized	RNFIS (proposed)
1	0.1704	0.1839	0.2366
2	0.0222	0.0180	0.0252
3	0.1649	0.1600	0.1509
…	…	…	…
99	0.0750	0.0937	0.0615
100	0.0888	0.0888	0.1471
MAP	0.1797	0.1851	0.2095

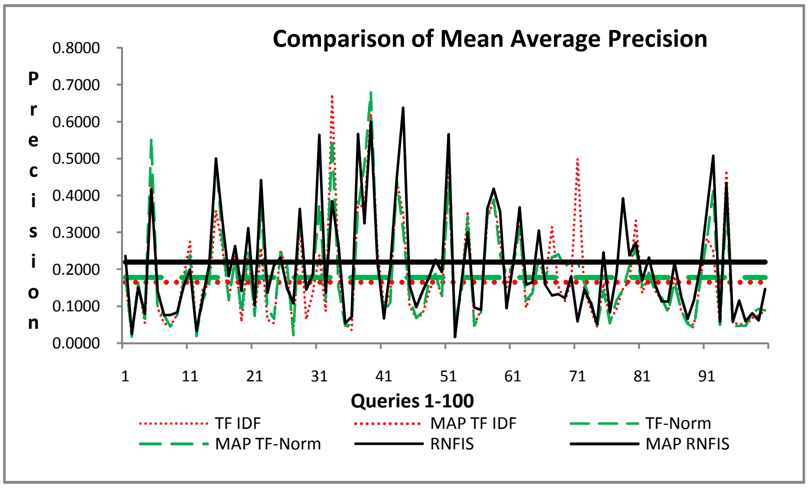
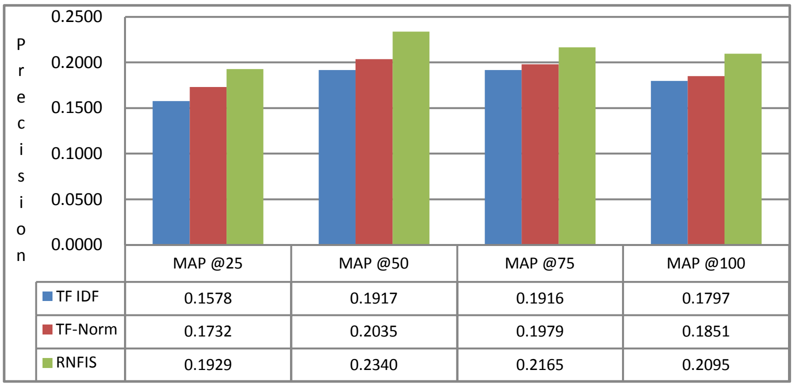


Fig. 4. Comparison of average of 100 cfqueries



	MAP @25	MAP @50	MAP @75	MAP @100
🟦 TF IDF	0.1578	0.1917	0.1916	0.1797
🟥 TF-Norm	0.1732	0.2035	0.1979	0.1851
🟩 RNFIS	0.1929	0.2340	0.2165	0.2095

Fig. 5. Comparison of MAP @ 25, 50, 75 & 100 cfqueries

4 Conclusion

In this research we introduced a comprehensive model called Ranked Neuro Fuzzy Inference System (RNFIS) for the retrieval of relevant documents. RNFIS calculates score of documents against a user query based on set of fuzzy rules. The RNFIS approach has given significant improvements in comparison with the performance of existing schemes such as length normalized TF and traditional TFIDF. The analysis of experimental results shows that RNFIS returns more relevant information by making synonym queries which cannot be retrieved by existing algorithms.

References

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press, New York (1999)
2. Raghavan, Schütze, H.: Introduction to Information Retrieval. Manning, Cambridge University Press, Cambridge (2008)
3. Rubens, N.O.: The application of fuzzy logic to the construction of the ranking function of information retrieval systems. Computer Modelling and New Technologies 10(1), 20–27 (2006)
4. Fuzzy logic Toolbox User's Guide (2004), `http://www.mathwork.com`, The MathWorks Inc. reterived on dated February 10 (2010)
5. Shaw, et al.: The Cystic Fibrosis Database: Content and Research Opportunities. LISR (13), 347–366 (1991)
6. Bordogna, G., Pasi, G.: Handling vagueness in information retrieval systems. In: Proceedings of the Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, November 20-23, pp. 110–114 (1995)
7. Baldwin, J.F., Lawry, J., Martin, T.P.: A Mass Assignment Theory of the Probability of Fuzzy Events. Fuzzy Sets and Systems (83), 353–367 (1996)
8. Baldwin, J.F., Martin, T.P., Pilsworth, B.W.: Fril Fuzzy and Evidential Reasoning in Artificial Intelligence. Research Studies Press Ltd., England (1995)
9. Stavrianou, A., Andritsos, P., Nicoloyannis, N.: Overview and Semantic Issues of Text Mining. SIGMOD Record 36(3) (September 2007)
10. Larocca Neto, J., Santos, A.D., Kaestner, C.A.A., Freitas, A.A.: Document Clustering and Text Summarization. In: Proceedings of the 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining (PADD 2000), pp. 41–55. The Practical Application Company, London (2000b)
11. Murad, M.A.A., Martin, T.: Similarity-Based Estimation for Document Summarization using Fuzzy Sets 1(4) (2005)
12. Baziz, M., et al.: A fuzzy set approach to concept-based information retrieval. In: EUSFLAT - LFA (2005)
13. Kraft, D.H., et al.: Vagueness and Uncertainty in Information Retrieval: How can Fuzzy Sets Help? In: Proceedings of the 2006 International Workshop on Research Issues in Digital Libraries, Kolkata, West Bengal, India (December 2006)
14. Lancaster, F.W., Fayen, E.G.: Information Retrieval Online. Melville Publishing Co., Los Angeles (1973)
15. Salton, G., Wong, A., Yang, C.S.: A vector space Model for Automatic Indexing. Communications of the ACM 18(11), 613–620 (1975)

The Study on Models Adjustment and Generation Capability of Artificial Neural Network

Shenglai Xia, Jingwu He, and Hongyu Chu

School of Aeronautic Science and Engineering,
Beijing University of Aeronautics and Astronautics,
37 Xueyuan Rd, Haidian District, 100191 Beijing, China
xiashenglai@ase.buaa.edu.cn

Abstract. Artificial neural network (ANN) is a thoroughly interdisciplinary area, covering neurosciences, physics, mathematics, economics and electronics. The applications of ANN are very diverse and effective. Some drawbacks, however, have been found accompanying with the applications of ANN. In order to overcome these drawbacks, many methods have been proposed. In this article, two issues will be referred, namely models adjustment and generalization capability of ANN. Models adjustment includes two aspects: the model's parameters adjustment and the model's architecture adjustment. The purpose of the former is to improve training speed, enhance convergence and stability of network. And the purpose of the latter is to enhance recognition ability. The model's architecture is adjusted through adding a binary-coding layer to it. In order to promote the generalization capability, the perfect training sample is put forward based on mathematics.

Keywords: ANN, parameters adjustment, architecture adjustment, generation capability.

1 Introduction

In recent years, ANN[1] has been a popular topic and made a great progress; furthermore, so many scientists are coming into this field annually. ANN is a thoroughly interdisciplinary area, covering neurosciences, physics, mathematics, economics and electronics. Today, ANN has found diverse applications in pattern recognition, signal processing, communication, control systems, optimization, and so on. For instance, Li[2] used ANN to identify the damage of Girder Bridge, Huang[3] adopted it to research anti-mechanical problem, while Qin[4] forecast the rainfall by this method and Zhang[5] predicted the daily water demands as well.

ANN is defined by processing elements, a topological structure, and learning rules. The processing elements are the basic operation elements of ANN. Learning rules determine how ANN acquires the pattern recognition knowledge and performs the classification accordingly.

© Springer-Verlag Berlin Heidelberg 2011

At present, Back Propagation[6-8] algorithm (BP) and its diversification forms are the most important and widely used ANN models. BP network is the core of feedforward network, and embodies the soul of ANN. It can mainly be applied for function approach, pattern recognition, classification and data compress.

BPANN is a multilayer feedforward network. The network architecture used by BPANN is formed by three or more layers--an input layer consisting of sample patterns of input data, an output layer storing the results of the decision-making process, and one or more hidden layers transferring the information in the network. The network learning process is a supervised training session. The general structure of such a network can be illustrated by Fig.1. The input is propagated through the network in a forward direction, on a layer-by-layer basis, to the output layer. The output layer is compared to target classification and the error is back propagated through the network layer-by-layer.

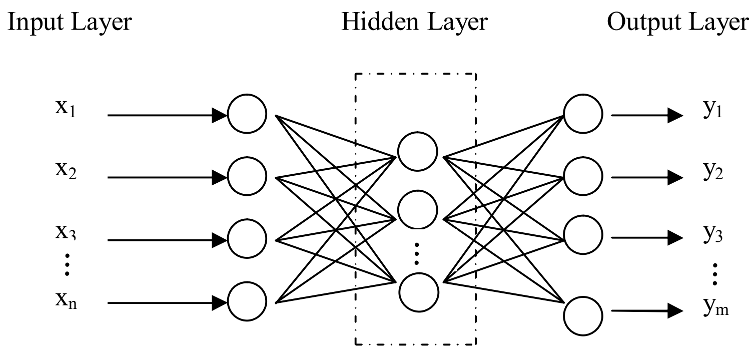


Fig. 1. A BP Neural Network Models

Although the BP algorithm remains as the most popular and effective way to train BPANN, there are some drawbacks accompanying it. These drawbacks include the following aspects:

① The training process is not convergent;

② The convergence speed is slow;

③ The training result is not convergent to the global optima;

④ When the input sample has some transformation, model's architecture is lack of recognition ability;

⑤ The training network is lack of generalization capability;

In view of these problems, in this paper, two issues will be proposed, namely models adjustment and generalization capability. Models adjustment includes two aspects. The first one is about model's parameters adjustment, and the other one is model's architecture adjustment.

2 Model's Parameters Adjustment

In order to resolve the problems mentioned above, firstly, the neuron which composes the network should be researched. A typical neuron in BPANN can be illustrated by Fig. 2.

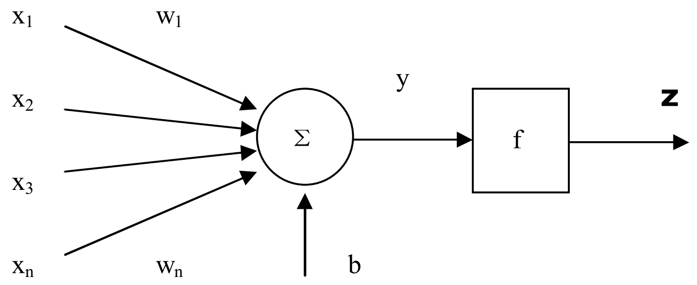


Fig. 2. A typical Neuron Models

The input-output relationship of the neuron can be described as

$$y = \sum_{i=1}^{n} w_i x_i + b \text{, } i\text{=1,2}\cdots\text{n} . \tag{1}$$

and

$$z = f(y) \ . \tag{2}$$

Where x_i*'s* are input to the neuron, w_i*'s* are corresponding connection weights, *b* is the bias, *y* is the net input to the neuron before activation, *z* is the output of the neuron, *f* is a nonlinear activation function.

When the entire network is concerned, the derivation of input-output relationship can be seen from many reference books. Here the process of derivation will be ignored. It can be found that many model's parameters can be adjusted. The main model's parameters include the number of layers, the number of hidden neurons, the learning rate, momentum term and expectation error. Many efforts have been made by the forerunner, and many effects have been acquired. These factors will be briefly discussed below.

① Number of Layers

The number of layers is an important factor for BP network. To increase the number of layer, the error will be declined and the precision will be increased. At the same time, the network will be complicated and the training time will be lengthened. Lippman[9] and Cyberko[10] put forward that two hidden layer network can resolve all the classifications, Nielson[11] thinks that one hidden layer network can resolve function approach so long as there is a larger number of hidden neurons.

② Number of Hidden Neurons

It is difficult to assure the number of hidden neurons. Initially, cut-and-try method was used to obtain the number and then many experiential formulas appeared later. Qin[4] put forward series of ANN.

③ Learning Rate

The learning rate decides how the weights will change. The weights come from circulation training progress. Choosing an appropriate learning rate parameter is a key factor in controlling the learning speed of the BP algorithm. The learning rate is a constant in a normal BPANN. However, during the different stages of the same learning process, the optimal value may be variable. Adjusting it automatically will be preferred during the learning progress.

④ Momentum Term

Because it is difficult to choose an appropriate learning rate parameter, a momentum term can be introduced to deal with this problem. The following formula can explain this relationship. That can be described as

$$w_{ji}(k+1) = w_{ji}(k) + \eta[(1-\alpha)D(k) + \alpha D(k-1)] \ . \tag{3}$$

Where w_{ji} is weight, η is learning rate, D is negative grads, α is the factor of momentum term. The value of α is between 0 and 1. Momentum term can reduce the oscillation of learning progress.

⑤ Expectation Error

The value of expectation error should be an appropriate value. In order to obtain the minor value, the number of hidden neuron and the training time will be increased. Generally, we can train several networks according to different value of expectation error, and then a better value will be obtained through comparing different results.

In addition, there are many other factors that can be adjusted, such as, activation function, optimization algorithm and so on. Nowadays, many researchers proposed self-adaptive method to adjust these factors, and they obtain many beneficial results.

3 Model's Architecture Adjustment

At first, four groups of data can be seen in the table 1. Each group of data expresses six points. It is difficult for us to find the relationship among these groups of data. However, if we depict the points on the picture, it is easy to find that each group data represents six vertexes of regular hexagon. Furthermore, it can be concluded that if the graph depicted by the first group of data is regarded as the original one, the pictures depicted via latter three groups of data are the transformations of the original one. They are the translation, rotation and similarity transformation respectively, which can be seen easily in Fig 3. The dotted line represents the original picture.

This transformation phenomenon can be easily found by our visual perception. But if ANN is used to recognize the transformation, maybe it is not so easy. The reason is that the ANN is on the basis of fixed and discrete retina theory. However, when we observe the object, our body can move freely and also our eyes and head can rotate

Table 1. Four Groups Data

Number	First Group	Second Group	Third Group	Fourth Group
1	(-3.54,-3.54)	(-1.04,0.16)	(-1.29,-4.83	(-2.00,0.00)
2	(-4.83,1.29)	(-2.33,4.99)	(-4.83,-1.29	(-1.00,1.73)
3	(-1.29,4.83)	(1.21,8.53)	(-3.54,3.54)	(1.00,1.73)
4	(3.54,3.54)	(6.04,7.24)	(1.29,4.83)	(2.00,0.00)
5	(4.83,-1.29)	(7.33,2.41)	(4.83,1.29)	(1.00,-1.73)
6	(1.29,-4.83)	(3.79,-1.13)	(3.54,-3.54)	(-1.00,-1.73)

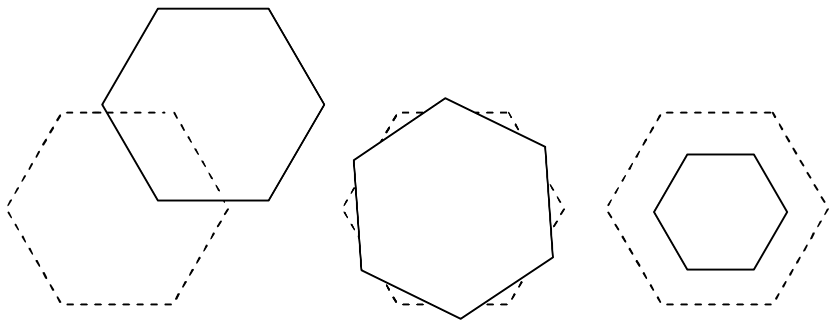


Fig. 3. Translation, Rotation and Similarity Transformation

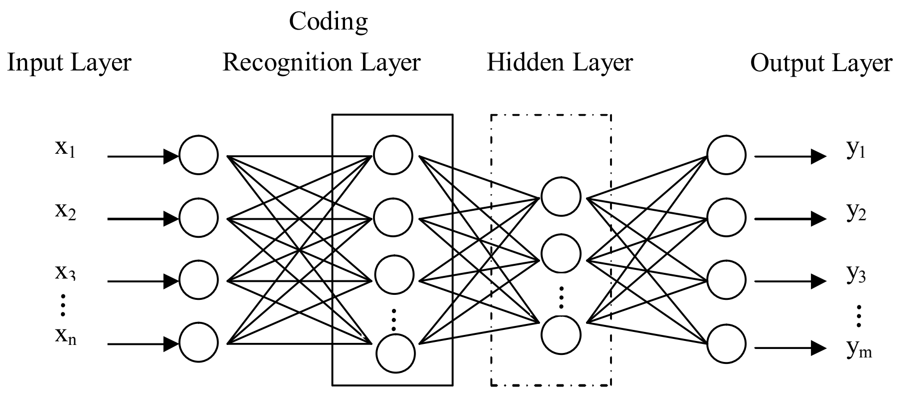


Fig. 4. New Network Models

permissively. Therefore, we hope that ANN can embrace the body's translation, eye and head rotation. The advantage is that if we have obtained a completed network, there is no necessity for us to train network again when we deal with the similar problems.

If these transformations are considered into the ANN, the model's architecture will be adjusted. New ANN architecture should be able to identify these changes. When the changes exist, the original training results can be applied instead of running the new network training.In order to achieve this, the network model needs to contain recognition function. The recognition outcome only has two keys: changes exist or not. According to this result, the binary-coding can fulfill the above requirement. 1 indicates these transformations are in existence and 0 indicates nonexistence. The new model's architecture can be seen in Fig.4.

4 Generalization Capabilities

The network generalization capability means that when the network has been trained, the network can make correct reactions to the sample that has not been trained. In other words, the ANN learning purpose is to search for the hidden orderliness of gross sample through finite training sample. The key of network generalization capability is how to select appropriate training sample. If the quantity of training sample is larger, the network will be over-fitting, on the contrary, if the quantity of training sample is fewer, the network will be lack-fitting. In regard to the research of train sample, many researchers have conducted a lot of investigation.

A perfect training sample should express all the information of the gross sample. Here I will research the training sample from the mathematical point of view. Mathematical model is shown in Fig. 5. ***G*** is the gross sample, ***T*** is the training sample. The ***T***-***G*** relationship can be described as

$\exists\, \boldsymbol{T}, \boldsymbol{G}, \boldsymbol{T} \subset \boldsymbol{G}$, $\boldsymbol{T}=[t_1,t_2\cdots, t_n]$, $t_1,t_2\cdots, t_n$ is linearly independent set of vectors, for y vector, $\forall\, y \in \boldsymbol{G}$ and $y \notin \boldsymbol{T}$, st.

$$y = \sum_{i=1}^{n} a_i t_i \, , a_i \in R^n \text{ and } i=1,2\cdots\text{n} . \tag{4}$$

Then, as a training sample, ***T*** covering all the information of the gross sample is a best training sample. That is to say, in the gross sample, $t_1,t_2\cdots, t_n$ is the maximal linearly independent vector group. Namely

$$\text{Rank } (\boldsymbol{T}) = \text{Rank } (\boldsymbol{G}) . \tag{5}$$

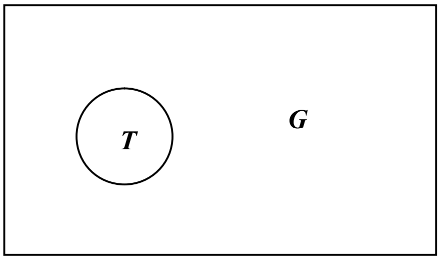


Fig. 5. Mathematical Model

Sometimes, not all the information is needed for some duty. So, the training sample quantity can be reduced, which can be referred from mathematics. Suppose there are two new vectors y' and t', representing available information respectively. The vectors y' and t' are extracted from y and t which are described above. The proposed method can be described clearly .Assume $\boldsymbol{W}_{n\times m}$ is a diagonal weight matrix,

$$\boldsymbol{W}_{n\times m}=[e_{ij}]_{n\times m},\ e_{ij}=0 \text{ or } 1 \text{ and } i=1,2\cdots n, j=1,2\cdots m\ . \tag{6}$$

Where 0 is unavailable information and 1 is available information.

$$t_i'=t_i\boldsymbol{W}_{n\times m}\ \ i=1,2\cdots n\ . \tag{7}$$

$$y' = \sum_{i=1}^{m} c_i t_i',\ c_i \in R^m \text{ and } i=1,2\cdots n\ . \tag{8}$$

The maximal linearly independent vectors group from a set of vectors t_i' composes a new matrix $\boldsymbol{D}$, the $\boldsymbol{D}$-$\boldsymbol{T}$ relationship can be described as

$$\text{Rank}\ (\boldsymbol{D}) \leqslant \text{Rank}\ (\boldsymbol{T})\ . \tag{9}$$

$$\boldsymbol{D} \subseteq \boldsymbol{T}\ . \tag{10}$$

5 Conclusions

Along with the continual adjustment of model's parameters and architecture, the drawbacks, such as convergence, training speed, local convergence, and stability of ANN, etc, will be eliminated gradually. At the same time, generalization capability of ANN will arouse more attention.

References

1. Niebur, D.: Introduction to Concepts in Artificial Neural Networks, pp. 1–12. NASA Jet Propulsion Laboratory, NASA (1995)
2. Li, Z.: Application of artificial neural network and wavelet analysis to damage identification of Girder Bridge. Doctoral dissertation. Southwest Jiao Tong University (2006) (in Chinese)
3. Huang, Y.: Anti-mechanical problems in the study of artificial neural network. Storage Transportation & Preservation of Commodities 30(6), 108–109 (2008)
4. Qin, G.: Artificial Neural Networks and Its Applications. Doctoral dissertation, Sichuan University (2003) (in Chinese)
5. Zhang, S.P., Watanabe, H., Yamada, R.: Prediction of daily water demands by neural networks. In: International Conference on Stochastic and Statistical Methods in Hydrology and Environment Engineering, Ontario, Canada, pp. 217–227 (1993)
6. May, J.A.: Using Artificial Neural Networks to Identify Unexploded Ordnance. Doctoral dissertation. Naval Postgraduate School, Monterey (1997)
7. Wilensky, G., Manukian, N., Neuhaus, J., et al.: Neural Network Studies. Technical report, Logicon R and Dassociates LOS Angeles CA (1993)

8. Torella, G., Lombardo, G.: Neural networks for the maintenance of aero engines. In: 31st AIAA/ASME/SAE/ASEE Joint Propulsion Conference, San Diego, CA (1995)
9. Lippmann, R.P.: An Introduction to computing with neural nets. IEEE ASSP Magazine 4(2), 4–22 (1987)
10. Cyberko, G.: Approximations by super positions of a sigmoidal function. Math. Control Signal System, 45–89 (1989)
11. Hecht-Nielsen, R.: Theory of back propagation neural network. Proc. of IJCNN 1, 593–603 (1989)

View Construction for Multi-view Semi-supervised Learning

Shiliang Sun, Feng Jin, and Wenting Tu

Department of Computer Science and Technology, East China Normal University
500 Dongchuan Road, Shanghai 200241, P.R. China
slsun@cs.ecnu.edu.cn, w.tingtu@gmail.com

Abstract. Recent developments on semi-supervised learning have witnessed the effectiveness of using multiple views, namely integrating multiple feature sets to design semi-supervised learning methods. However, the so-called multi-view semi-supervised learning methods require the availability of multiple views. For many problems, there are no ready multiple views, and although the random split of the original feature sets can generate multiple views, it is definitely not the most effective approach for view construction. In this paper, we propose a feature selection approach to construct multiple views by means of genetic algorithms. Genetic algorithms are used to find promising feature subsets, two of which having maximum classification agreements are then retained as the best views constructed from the original feature set. Besides conducting experiments with single-task support vector machine (SVM) classifiers, we also apply multi-task SVM classifiers to the multi-view semi-supervised learning problem. The experiments validate the effectiveness of the proposed view construction method.

1 Introduction

Semi-supervised learning, aiming to improve the performance of pattern recognition systems using both labeled and unlabeled data, is an active research direction [1]. In particular, there is a family of methods making use of multiple feature sets to design semi-supervised learning methods. These multiple feature sets are often called multiple views and the corresponding approaches are referred to as multi-view semi-supervised learning methods.

For some domains, multiple views are readily available. For example, in multimedia information processing we can treat video and audio signals as two distinct views. In document classification, characters and images can serve as two views. However, for many other domains, there are no natural multiple views available. Although we can split the original feature set at random into different views, there is no guarantee to obtain a satisfactory result by this approach.

This paper focuses on the view construction problem to benefit multi-view semi-supervised learning methods. Genetic algorithms (GAs) [2] are employed to select candidate feature subsets which are further assessed by their agreements for classification. The feature subsets finally selected are then regarded as multiple views, which are subsequently used by multi-view semi-supervised learning methods. The feasibility of the proposed approach is evaluated on multiple experiments involving the traditional

© Springer-Verlag Berlin Heidelberg 2011

single-task learning and recent multi-task learning paradigms where co-training [3] is taken as the specific multi-view semi-supervised learning method.

In the rest of this paper, after giving a brief review of multi-view semi-supervised learning methods in Section 2, we describe our view construction method in Section 3. Experimental results are reported in Section 4 and conclusions are drawn in Section 5.

2 Multi-view Semi-supervised Learning

During the development of multi-view semi-supervised learning, co-training [3] is known as an important pioneering method. It trains two classifiers respectively from two views. Then the classifiers iteratively use their confident predictions on the unlabeled data to enlarge the labeled set. The effectiveness of co-training has been validated by several studies [3,4].

Subsequently, by probabilistically labeling the whole unlabeled data pool, Nigam and Ghani [5] proposed a variant of co-training named co-EM. The idea of co-training has also been extended to semi-supervised learning methods without multiple views such as tri-training [6] and democratic co-learning [7]. Later, Balcan et al. [8] relaxed the conditional independence assumption in co-training with a much weaker expansion condition. The recently proposed manifold co-regularization method [9] can be regarded as a combination of co-training and manifold regularization.

3 View Construction

3.1 Feature Selection Using GAs

The goal of our feature subset selection is to use less features to get the same or better performance on a separate validation set. For this purpose, we use GAs. The chromosomes in GAs are binary bit strings whose lengths represent the numbers of features, and each bit is associated with one feature. If the ith bit is 1, the ith feature is selected, and otherwise this feature is ignored.

The initial population of chromosomes are generated by randomly flipping each binary bit. Our individual selection strategy is cross generational. Suppose the size of the population is n. The size of the offspring will double this number. We then select the best n individuals from the combined parent-offspring population as the next generation. The crossover and mutation probability used in the experiments are 0.66 and 0.03, respectively.

3.2 Determining Two Views

After the running of GAs, each individual in the final genetic population corresponds to a candidate feature subset. As the number of feature subsets is the same as the final population size, we can generate many views. However, in the present study, we only consider generating two views for the sake of co-training. To this end, we evaluate the feature subsets on a validation set, and select the pair with the largest number of classification agreements as the two views.

Consequently, two views are constructed by the feature selection and validation procedures. As features having significant contributions to the classification task are selected through the feature selection procedure, the two views would be very helpful to improve the performance of subsequent multi-view semi-supervised learning methods.

3.3 Multi-task Learning

Generally, multi-task learning is motivated by the fact that learning multiple related tasks simultaneously can be advantageous in terms of predictive performance relative to learning these tasks independently [10,11]. It is thus advantageous to evaluate our view construction method on multi-task classifiers in addition to traditional single-task classifiers. To this end, here we briefly introduce the adopted multi-task support vector machines (SVMs), which was proposed by Evegniou et al. [12].

Suppose we have T learning tasks and all data for the tasks are from the same space $X \times Y$ where $X \in R^d$ and $Y \in R$. For each task we have m training examples $\{(x_{1t}, y_{1t}), (x_{2t}, y_{2t}), \ldots, (x_{mt}, y_{mt})\}$ sampled from a distribution P_t on $X \times Y$. P_t's are different but related among tasks [12]. The goal is to learn T functions $f_1, f_2, \ldots, f_T$ such that $f_t(x_{it}) \approx y_{it}$. The case $T = 1$ corresponds to the traditional single-task learning diagram.

In this paper, we assume that function f_t for task t is a hyperplane, that is $f_t(x) = w_t \cdot x$. Binary classification is considered where y_{it} takes the values ± 1, and the decision on a test example is the sign of $w_t \cdot x$.

According to Evegniou et al. [12], weights w_t are formulated as $w_t = w_0 + v_t$, where vectors v_t are small when tasks are similar to each other. Therefore, the following optimization problem will be solved:

$$\min_{w_0, v_t, \xi_{it}} J = \sum_{t=1}^{T} \sum_{i=1}^{m} \xi_{it} + \frac{\lambda_1}{T} \|v_t\|^2 + \lambda_2 \|w_0\|^2$$
$$\text{s.t.} \quad \begin{cases} y_{it}(w_0 + v_t)^\top x_{it} \geq 1 - \xi_{it}, \\ \xi_{it} \geq 0, \quad i = 1, \ldots, m; t = 1, \ldots, T \end{cases} \tag{1}$$

where λ_1 and λ_2 are positive regularization coefficients, and ξ_{it} are slack variables measuring the errors that models w_t commit on the training data.

4 Experiments

The feasibility of the proposed view construction method is evaluated on two sets of multi-view semi-supervised learning experiments. The first one is a single-task learning scenario. The second one combines multi-task learning with multi-view semi-supervised learning.

For all the experiments, ten-fold cross-validation (CV) is adopted to evaluate the performance, and the average accuracies across five random runs of this CV are reported. Linear SVMs are adopted as the specific classifiers, which have a natural expression of classification confidence in terms of the margin between an example and the classification hyperplane. The number of iterations in co-training is set to 25. Since there are two classifiers respectively built from two views in the process of co-training iterations [3], we report classification accuracies of these two classifiers after every iteration. The random split approach to generate multiple views is employed as a baseline and compared to our proposed view construction method.

4.1 Single-Task Classifiers

4.1.1 Handwritten Digit Recognition

Firstly, we performed a handwritten digit recognition experiment using the United State Postal Service (USPS) database. The data include ten digits from 0 to 9 but only one view. Two figures 0 and 3 are chosen for performance evaluation, as they have some overlapping in shape which makes accurate classification not trivial. Thus the data used have 1553 digits 0 and 824 digits 3 (positive examples).

In order to use the multi-view semi-supervised learning method co-training, we create two views from the given data. The proposed method is first using GAs for feature selection to generate many candidate views. Then after the GAs, two feature subsets which have the largest number of classification agreements are selected as the final two views. Here, the two views chosen have 66 and 75 features, respectively. The compared method is a random split of the original features. It means that two views respectively have 128 and 128 features are generated, by randomly dividing the original 256 features.

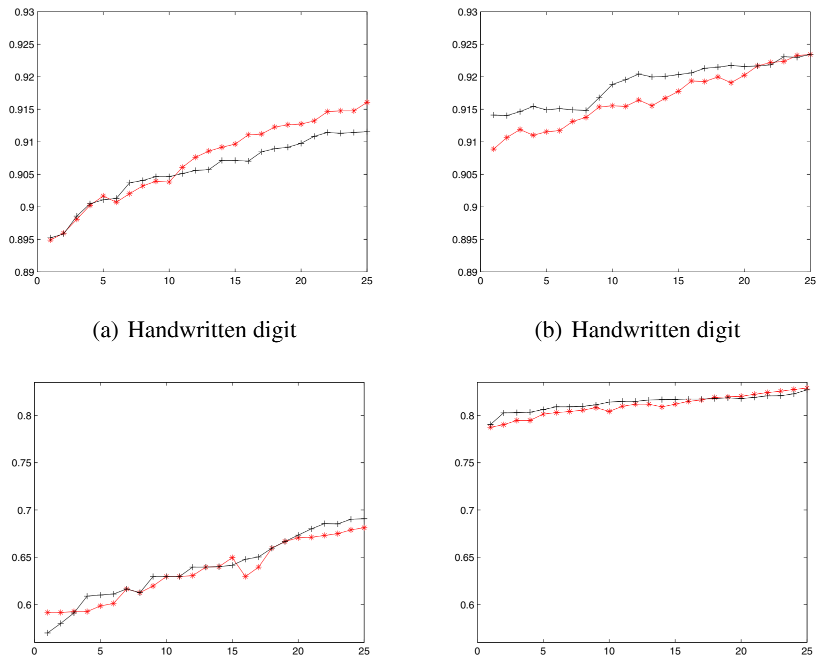
After multiple views are formed, co-training can work for multi-view semi-supervised learning. The ten-fold CV is performed as follows. Fifteen positive and thirty negative examples are randomly picked out to be the labeled training set. During each round of co-training, one positive and two negative examples are found with high confidence and added to the labeled training set by each classifier from each view. This step will not terminate until the maximum iteration number is reached. At the end of each iteration, two SVM classifiers from the two views are trained and then classify the test fold.

The test accuracy rates averaged over five such CVs are shown in Figure 1(a) and Figure 1(b), where the two methods have the same parameter setting. It is clear that for all the two classifiers the predictive accuracies of the proposed method are always better than that of the random split method with the same number of iterations.

4.1.2 Heart Classification

To better understand the performance of the proposed method, we continue our investigation using heart classification data from the UCI data repository. The data set contains 150 positive and 120 negative examples. It is a single-view and two-class classification data set. The total number of features is 13. Two views respectively with two and three features are generated by the proposed method; while for the random split, the two views respectively have six and seven dimensions.

For co-training, the original labeled training set consists of 15 positive and 12 negative examples. During each iteration, two positive and one negative example are found and added to the labeled training set. Other experimental settings are the same with the above experiment. The classification results are provided in Figure 1(c) and Figure 1(d), where the proposed method greatly outperforms the random split method.



(a) Handwritten digit (b) Handwritten digit

(c) Heart classification (d) Heart classification

Fig. 1. Accuracies of co-training with single-task SVMs on different data sets. Two views are generated by random split (a, c) or our proposed method (b, d).

4.2 Multi-task Classifiers

4.2.1 Synthetic Data

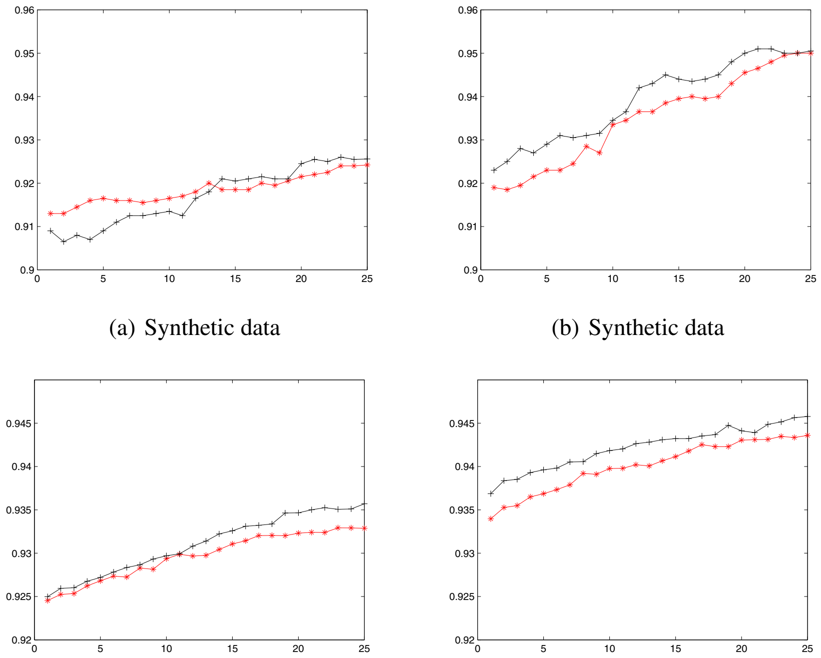
Here we combine multi-task learning with multi-view semi-supervised learning to evaluate our proposed view construction method. We first test the performance of the method on a synthetic data set.

The case of two-class classification is considered. The number of tasks is $T = 20$. Each of the w_t parameters of these tasks is selected from a 20-dimensional Gaussian distribution with zero mean and diagonal covariance matrix Cov $= diag(1, 0.95, 0.90, \ldots, 0.15, 0.10, 0.05)$. For each of the 20 tasks, we generate 20 examples. As a result, the total number of examples for the 20 tasks is 400.

Our view construction method found two views respectively with six and seven dimensions, while each view found by the random split has 10 dimensions. The final experimental results are given in Figure 2(a) and Figure 2(b) where for both the two classifiers the proposed method outperforms the random split method during the 25 rounds. The efficacy of the proposed method is validated again by this experiment.

4.2.2 Real-World Data

The handwritten digit recognition database of USPS mentioned before is used but with all the ten digits. The ten digits in this data set are divided into five groups, and each



(a) Synthetic data (b) Synthetic data

(c) Handwritten digit (d) Handwritten digit

Fig. 2. Accuracies of co-training with multi-task SVMs on different data sets. Two views are generated by random split (a, c) or our proposed method (b, d).

group contains one even and one odd digits (in this paper we do not consider the problem of how to group tasks optimally). Hence, there are five tasks to learn and each task is to classify the corresponding even and odd digits.

The number of features obtained by the proposed view construction method for two view are 66 and 67, respectively. Classification results during the iterations of co-training are shown in Figure 2(c) and Figure 2(d), from which we see that the views constructed by the proposed method are more suitable than random split for the current multi-view classification problem.

The experimental results in Section 4.2.1 and 4.2.2 indicate that the features selected through the view construction method are also suitable for multi-view semi-supervised learning combined with multi-task learning.

5 Conclusions

This paper proposed a view construction method for multi-view semi-supervised learning. It uses GAs to select candidate feature subsets from which multiple views are determined by investigating their classification agreements. We have reported experimental results on different data sets with both single-task and multi-task SVM classifiers, which indicate that the new method can lead to significant performance improvements compared to the random feature split approach adopted previously by others.

In the future, studies on the theoretical justification for the proposed method and its extensions to other multi-view learning paradigms such as multi-view active learning are worthy pursuing.

Acknowledgments. This work is supported in part by the National Natural Science Foundation of China under Project 61075005, 2011 Shanghai Rising-Star Program, and the Fundamental Research Funds for the Central Universities.

References

1. Chapelle, O., Schölkopf, B., Zien, A.: Semi-supervised Learning. MIT Press, Cambridge (2006)
2. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1998)
3. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the 11th Annual Conference on Computational Learning Theory, pp. 92–100 (1998)
4. Sun, S.: Semantic features for multi-view semi-supervised and active learning of text classification. In: Proceedings of the IEEE International Conference on Data Mining Workshops, pp. 731–735 (2008)
5. Nigam, K., Ghani, R.: Analyzing the effective and applicability of co-training. In: Proceedings of the 9th International Conference on Information and Knowledge Management, pp. 86–93 (2000)
6. Zhou, Z., Li, M.: Tri-training: Exploiting unlabeled data using three classifiers. IEEE Transactions on Knowledge and Data Engineering 17(11), 1529–1541 (2005)
7. Zhou, Y., Goldman, S.: Democratic co-learning. In: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, pp. 594–602 (2004)
8. Balcan, M., Blum, A., Yang, K.: Co-training and expansion: Towards bridging theory and practice. Advances in Neural Information Processing Systems 17, 89–96 (2005)
9. Sindhwani, V., Rosenberg, D.: An RKHS for multiview learning and manifold co-regularization. In: Proceedings of the 25th International Conference on Machine Learning, pp. 976–983 (2008)
10. Caruana, R.: Multitask learning. Machine Learning 28(1), 41–75 (1997)
11. Jin, F., Sun, S.: A multitask learning approach to face recognition based on neural networks. In: Fyfe, C., Kim, D., Lee, S.-Y., Yin, H. (eds.) IDEAL 2008. LNCS, vol. 5326, pp. 24–31. Springer, Heidelberg (2008)
12. Evgeniou, T., Pontil, M.: Regularized multi-task learning. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 109–117 (2004)

Fast Learning Fully Complex-Valued Classifiers for Real-Valued Classification Problems

R. Savitha[1], S. Suresh[2], N. Sundararajan[1], and H.J. Kim[3]

[1] School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore
[2] School of Computer Engineering, Nanyang Technological University, Singapore
[3] CIST, Korea University, Seoul

Abstract. In this paper, we present two fast learning neural network classifiers with a single hidden layer: the 'Phase Encoded Complex-valued Extreme Learning Machine (PE-CELM)' and the 'Bilinear Branch-cut Complex-valued Extreme Learning Machine (BB-CELM)'. The proposed classifiers use the phase encoded transformation and the bilinear transformation with a branch-cut at 2π as the activation functions in the input layer to map the real-valued features to the complex domain. The neurons in the hidden layer employ the fully complex-valued activation function of the type of a hyperbolic secant function. The parameters of the hidden layer are chosen randomly and the output weights are estimated as the minimum norm least square solution to a set of linear equations. The classification ability of these classifiers are evaluated using a set of benchmark data sets from the UCI machine learning repository. Results highlight the superior classification ability of these classifiers with least computational effort.

1 Introduction

Recently, complex-valued neural networks are gaining attention to approximate phase accurately [1–5]. Complex-valued neural networks have better computational ability than real-valued networks [6]. Their inherent orthogonal decision boundary [7] that provides them with an exceptional ability to perform real-valued classification tasks [8] motivates researchers to develop complex-valued classifiers.

Complex-valued classifiers available in the literature include the Multi Layer Multi Valued Network (MLMVN) [9] and the single layer complex-valued neural network with phase encoded input features [10] referred to as 'Phase Encoded Complex Valued Neural Network (PE-CVNN)'. A multi-valued neuron used in the MLMVN [9] uses multiple-valued threshold logic to map the complex-valued input to C discrete outputs using a piecewise continuous activation function, where C is the total number of classes. The transformation used in the MLMVN is not unique, leading to misclassification. The misclassification is further enhanced by the increase in number of sectors inside the unit circle in multi-category classification problems with more number of classes (C). Moreover, the

© Springer-Verlag Berlin Heidelberg 2011

MLMVN uses a derivative-free global error correcting rule for network parameter update that requires significant computational effort.

In the PE-CVNN presented in [10], [11], the real-valued input feature is phase encoded in $[0, \pi]$ to obtain the complex-valued input feature. This transformation retains the relational property and spatial relationship among the real-valued input features [10]. However, the activation functions used in [10], [11] are similar to the split complex-valued activation functions and do not preserve the phase information of the error signal during the backward computation. This might result in inaccurate estimation of the decision function while performing classification problems. In addition, the gradient descent based learning, presented in [10], requires significant time to train the classifier.

In this paper, we propose two complex-valued classifiers that provide better generalization ability and require lesser computational effort. The classifiers have a non-linear input/hidden layer and a linear output layer. The neurons at the input layer transform the real-valued input features to the complex domain ($\Re \rightarrow \mathbb{C}$). The bilinear transformation [9] with a branch-cut (δ) and the phase encoding transformation presented in [10] are used as the transformations in the input layer. The network employing the bilinear transformation with a branch-cut in the input layer is referred to as 'Bilinear Branch-cut Complex-valued Extreme Learning Machine (BB-CELM)' and the network employing the phase encoding transformation in the input layer is referred to as 'Phase Encoded Complex-valued Extreme Learning Machine (PE-CELM)'. The neurons in the hidden layer of these networks use the fully complex-valued activation function of the type of a hyperbolic secant function [4]. Similar to the C-ELM [2], the parameters of the hidden neurons are chosen randomly and parameters of the output neurons of the network are estimated analytically.

The performances of both the BB-CELM and the PE-CELM are studied in comparison with other complex-valued and a few real-valued classifiers on three multi-category benchmark classification problems from the UCI repository [12]. It will be observed from the performance study that the proposed classifiers outperform the other classifiers available in the literature.

The paper will be organized as follows: Section 2 presents the detailed description of the proposed classifiers. Section 3 presents the performance results of these classifiers in comparison with other classifiers on a set of multi-category benchmark classification problems. Finally, Section 4 summarizes the conclusion from the study.

2 Description of the Classifiers

In this section, we present the detailed description of two fast learning classifiers, the Bilinear Branch-cut Complex-valued Extreme Learning Machine (BB-CELM) and the Phase Encoded Complex-valued Extreme Learning Machine (PE-CELM) in the complex-domain.

2.1 Classification Problem Definition

Let us assume N random observations $\{(\mathbf{x}_1, c_1), \cdots, (\mathbf{x}_t, c_t), \cdots, (\mathbf{x}_N, c_N)\}$, where $\mathbf{x}_t \in \Re^m$ are the m-dimensional real-valued input features of tth observation and $c_t \in \{1, 2, \cdots, C\}$ is its class label. The coded class label in the complex domain $\mathbf{y}^t$ are obtained using:

$$y_l^t = \begin{cases} 1+i, & \text{if } c_t = l, \\ -1-i, & \text{otherwise,} \end{cases} \quad l = 1, 2, \cdots, C \tag{1}$$

Now, the classification problem in the complex domain can be viewed as finding the decision function (F) that maps the real-valued input features to the complex-valued coded class labels, i.e., $F : \Re^m \rightarrow \mathbb{C}^C$, and then predicting the class labels of new, unseen samples with certain accuracy.

2.2 Fast Learning Complex-Valued Classifiers

The basic building block of the PE-CELM and the BB-CELM classifiers is the complex-valued extreme learning machine [2]. The PE-CELM and BB-CELM are the single hidden layer networks, with a non-linear input/hidden layer and a linear output layer as shown in Fig. 1.

The neurons in the input layer employ these transformations to map the real-valued input features to the complex domain ($\Re \rightarrow \mathbb{C}$):

– **The bilinear transformation with a branch cut:** As the bilinear transformation [9] results in aliasing at 0 and 2π, we introduce a branch cut around 2π. The transformation thus obtained is termed as a bilinear transformation with a branch cut. The network using this transformation (Eq. (2)) at the input layer is referred to as 'Bilinear Branch-cut Complex-valued Extreme Learning Machine (BB-CELM)'.

$$z_l = exp(i(2\pi x_l - \delta)); \ l = 1, \cdots, m \tag{2}$$

– **The phase encoded transformation [10]:** The network with the phase encoded transformation (Eq. (3)) at the input layer is called 'Phase Encoded Complex-valued Extreme Learning Machine (PE-CELM)'.[1]

$$z_l = exp(i\pi x_l); \ l = 1, \cdots, m \tag{3}$$

The neurons in the hidden layer of the BB-CELM/PE-CELM classifiers employ the fully complex-valued activation function of the type of a hyperbolic secant function [4], and their responses are given by

$$h_j = sech\left(\mathbf{u}_j^T (\mathbf{z}_t - \mathbf{v}_j)\right); j = 1 \cdots K \tag{4}$$

where $\mathbf{u}_j$ is the complex-valued scaling factor and $\mathbf{v}_j$ is the center of the j-th neuron, and $sech(x) = 2/(e^x + e^{-x})$.

[1] It must be noted here that all the input features are scaled in $[0, 1]$, *i.e.,* $\mathbf{x}_t \in \Re[0, 1], \ t = 1 \cdots N$.

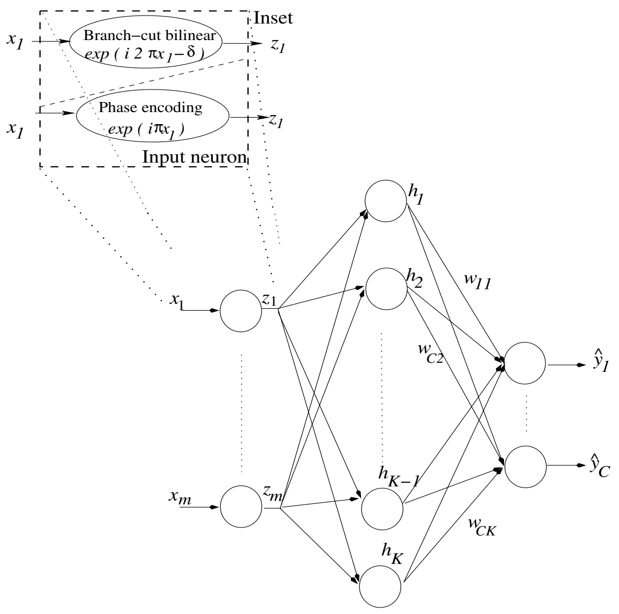


Fig. 1. The architecture of a BB-CELM/PE-CELM. The transformations are shown in the inset.

The output ($\widehat{\mathbf{y}}$) of the classifiers is given by:

$$\widehat{y}_l = \sum_{j=1}^{K} w_{lj} h_j \tag{5}$$

where w_{lj} are the complex-valued weight connecting the l-th output neuron and the j-th hidden neuron.

Eq. (5) can be written in a matrix form as

$$\widehat{Y} = WH \tag{6}$$

where W is the matrix of all output weights connecting the hidden layer, and H is the $K \times N$ hidden layer response matrix and is given by

$$H(V,B,Z) = \begin{bmatrix} sech\left(\mathbf{u}_1^T \|\mathbf{z}_1 - \mathbf{v}_1\|\right) & \cdots & sech\left(\mathbf{u}_1^T \|\mathbf{z}_N - \mathbf{v}_1\|\right) \\ \vdots & \vdots & \vdots \\ sech\left(\mathbf{u}_K^T \|\mathbf{z}_1 - \mathbf{v}_K\|\right) & \cdots & sech\left(\mathbf{u}_K^T \|\mathbf{z}_N - \mathbf{v}_K\|\right) \end{bmatrix} \tag{7}$$

Similar to the C-ELM [2], the parameters of the hidden neurons ($\mathbf{u}_j$, $\mathbf{v}_j$) are chosen randomly and the output weights W are estimated by the least squares method according to:

$$W = YH^{\dagger} \tag{8}$$

where $H^{\dagger}$ is the Moore-Penrose pseudo-inverse of the hidden layer output matrix, and Y is the complex-valued coded class label.

The class labels can be estimated from the outputs using:

$$\widehat{c} = \max_{l=1,2,\cdots,C} \text{ real } (\widehat{y}_l) \tag{9}$$

The proposed PE-CELM/BB-CELM algorithm can be summarized as:

– For a given training set (X, Y), select the appropriate number of hidden neurons K.
– Choose the scaling factor U and the neuron centers V randomly.
– Calculate the output weights W analytically: $W = TY_K^{\dagger}$.

3 Performance Evaluation

In this section, the performances of the PE-CELM and the BB-CELM are evaluated on a set of benchmark multi-category classification problems from the UCI machine learning repository [12]. The performance results are compared with other complex-valued classifiers, viz., MLMVN [9] and PE-CVNN [11]. To highlight the advantages of the orthogonal decision boundaries, the performances are also compared with a few real-valued classifiers, viz., the Minimal Resource Allocation Network (MRAN) [14], the Growing and Pruning Radial Basis Function Network (GAP-RBFN) [15], the Online Sequential Extreme Learning Machine (OS-ELM) [16], the Support Vector Machines (SVM) [17], the Self-regulatory Resource Allocation Network (SRAN) [18], and the Real Coded Genetic Algorithm Extreme Learning Machines (RCGA-ELM) [19].

The average (η_a) (Eq. (10)) and over-all (η_o) (Eq. (11)) classification efficiencies [20] of the classifiers derived from their confusion matrices are used as the performance measures for comparison in this study.

$$\eta_a = \frac{1}{C}\sum_{i=1}^{C}\frac{q_{ii}}{N_i} \times 100\% \tag{10}$$

$$\eta_o = \frac{\sum_{i=1}^{C} q_{ii}}{\sum_{i=1}^{C} N_i} \times 100\% \tag{11}$$

where q_{ii} is the total number of correctly classified samples in the class c_i and N_i is the total number of samples belonging to a class c_i in the data set. In this paper, an appropriate number of hidden neurons is selected using the addition/deletion of neurons to obtain an optimal performance, as discussed in [13] for real-valued networks.

The number of classes, the number of features, the size of the training and the testing data sets for the three multi-category benchmark classification problems considered for the study, are presented in Table 1. The image segmentation data set is a well-balanced data set. The data set for the glass identification

Table 1. Description of benchmark data sets selected from [12] for performance study

Problem	No. of features	No. of classes	No. of samples	
			Training	Testing
Image Segmentation (IS)	19	7	210	2,100
Vehicle Classification (VC)	18	4	424	422
Glass Identification (GI)	9	6	109	105

Table 2. Performance results for benchmark multi-category classification problems

Dom -ain	Algo.	Image Segmentation				Vehicle Classification				Glass Identification			
		K	Time (sec.)	Testing		K	Time (sec.)	Testing		K	Time (sec.)	Testing	
				η_o	η_a			η_o	η_a			η_o	η_a
Real	MRAN	76	783	86.52	86.52	100	520	59.94	59.83	51	520	63.81	70.24
	GAP-RBF	83	365	87.19	87.19	81	452	59.24	58.23	75	410	58.29	72.41
	OS-ELM	100	21	90.67	90.67	300	36	68.95	67.56	60	15	67.62	70.12
	SVM[a]	96	721	90.62	90.62	234	550	68.72	67.99	102	320	64.23	60.01
	SMC-RBF	43	142	91	91	75	120	74.18	73.52	58	97	78.09	77.96
	RCGA-ELM	50	-	91	91	75	-	74.2	74.4	60	-	78.1	-
	SRAN	47	22	92.3	92.3	55	113	75.12	76.86	59	28	86.21	80.95
Comp.	MLMVN	80	1384	83	83	90	1396	78	77.25	85	1421	73.24	66.83
	PE-CVNN	-	-	93.2[b]	-	-	-	78.7[a]	-	-	-	65.5[a]	-
	BB-CELM	**65**	**0.03**	**92.5**	**92.5**	**100**	**0.11**	**80.3**	**80.4**	**70**	**0.08**	**88.16**	**81**
	PE-CELM	**75**	**0.03**	**92.1**	**92.1**	**100**	**0.11**	**80.8**	**81.1**	**70**	**0.08**	**86.35**	**80**

[a] Support vectors

[b] -A single layer network was used in [11]. Also, in [11], a 10-fold validation has been done using 90% of the total samples in training and the remaining 10% for testing in each validation. Training/testing samples used in our work are shown in Table 1.

problem is a highly unbalanced data set, with no samples in one class. The vehicle classification data set is also not a well-balanced data set.

The number of hidden neurons chosen according to the discussion in Section 2.2, the training time and the testing performance measures of the BB-CELM and the PE-CELM classifiers, in comparison with other complex-valued and a few real-valued classifiers, are presented in Table 2. The results for the SRAN and RCGA-ELM is reported from [18] and [19], respectively. For the other real-valued classifiers, the results are reproduced from [20]. The results for the PE-CVNN are reproduced from [11] (single layered network) and those of the MLMVN are generated using the toolbox available in the author's web site[1].

From the results, it can be observed that the BB-CELM and PE-CELM classifiers outperform other complex-valued classifiers. The classification performances

[1] `http://www.eagle.tamut.edu/faculty/igor/Downloads.htm`

of the PE-CVNN and the MLMVN classifiers are affected by their activation functions. The activation function of the PE-CVNN is similar to the split complex-valued activation function that does not exploit the complete advantage of the orthogonal decision boundaries of the complex-valued neural networks. On the other hand, the MLMVN maps the complex-valued inputs to C discrete outputs in the unit circle, which increases with C and affects the classification performance.

It can be also observed that the proposed classifiers perform better than the real-valued classifiers, especially in the classification of the unbalanced data sets. This is because of the orthogonal decision boundaries, which is inherent in complex-valued neural networks. It can also be observed that the BB-CELM and the PE-CELM require significantly lesser computational effort, compared to all the other classifiers performing the classification task.

Comparing the performances of the PE-CELM and the BB-CELM classifiers, it is observable that the BB-CELM classifier performs better than the PE-CELM classifier. The phase encoding transformation used in the PE-CELM classifier maps the real-valued input features to only two quadrants of the unit circle prohibiting it from exploiting the advantages of the orthogonal decision boundaries completely. The bilinear transformation with a branch cut around 2π used in the BB-CELM classifier uses all the four quadrants of the complex plane.

4 Conclusion

In this paper, we present two efficient, fast learning complex-valued classifiers, the Bilinear Branch-cut Complex-valued Extreme Learning Machine (BB-CELM) and the Phase Encoded Extreme Learning Machine (PE-CELM). It is a single hidden layer neural network with randomly selected input parameters and analytically estimated output parameters. The performances of the BB-CELM and the PE-CELM are evaluated using three benchmark classification problems. From the performance study, it is evident that both the proposed classifiers outperform other classifiers with lesser computational effort.

Acknowledgement. The authors would like to thank the MOE, Singapore for their financial support through the TIER I grant to conduct the study.

References

1. Kim, T., Adali, T.: Fully-complex multilayer perceptron network for nonlinear signal processing. Journal of VLSI Signal Processing 32(1-2), 29–43 (2002)
2. Li, M.B., Huang, G.B., Saratchandran, P., Sundararajan, N.: Fully complex extreme learning machines. Neurocomputing 68(1-4), 306–314 (2005)
3. Savitha, R., Suresh, S., Sundararajan, N., Saratchandran, P.: A new learning algorithm with logarithmic performance index for complex-valued neural networks. Neurocomputing 72(16-18), 3771–3781 (2009)
4. Savitha, R., Suresh, S., Sundararajan, N.: A fully complex-valued radial basis function network and its learning algorithm. International Journal of Neural Systems 19(4), 253–267 (2009)

5. Savitha, R., Suresh, S., Sundararajan, N.: A self-regulated learning in fully complex-valued radial basis function networks. In: Proc. of International Joint Conference on Neural Networks, IJCNN 2010 (2010)
6. Nitta, T.: The computational power of complex-valued neuron. In: Kaynak, O., Alpaydın, E., Oja, E., Xu, L. (eds.) ICANN 2003 and ICONIP 2003. LNCS, vol. 2714, pp. 993–1000. Springer, Heidelberg (2003)
7. Nitta, T.: On the inherent property of the decision boundary in complex-valued neural networks. Neurocomputing 50, 291–303 (2003)
8. Nitta, T.: Solving the XOR problem and the detection of symmetry using a single complex-valued neuron. Neural Networks 16(8), 1101–1105 (2003)
9. Aizenberg, I., Moraga, C.: Multilayer feedforward neural network based on multi-valued neurons (MLMVN) and a backpropagation learning algorithm. Soft Computing 11(2), 169–193 (2007)
10. Amin, M.F., Murase, K.: Single-layered complex-valued neural network for real-valued classification problems. Neurocomputing 72(4-6), 945–955 (2009)
11. Amin, M.F., Islam, M.M., Murase, K.: Ensemble of single-layered complex-valued neural networks for classification tasks. Neurocomputing 72(10-12), 2227–2234 (2009)
12. Blake, C., Merz, C.: UCI Repository of Machine Learning Databases, Department of Information and Computer Sciences, University of California, Irvine (1998), http://archive.ics.uci.edu/ml/
13. Suresh, S., Omkar, S.N., Mani, V., Guru Prakash, T.N.: Lift coefficient prediction at high angle of attack using recurrent neural network. Aerospace Science and Technology 7(8), 595–602 (2003)
14. Lu, Y., Sundararajan, N., Saratchandran, P.: A sequential learning scheme for function approximation using minimal radial basis function neural networks. Neural Computation 9(2), 461–478 (1997)
15. Huang, G.-B., Saratchandran, P., Sundararajan, N.: An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks. IEEE Transactions on System, Man and Cybernetics B 34(6), 2284–2292 (2004)
16. Liang, N.-Y., Huang, G.-B., Saratchandran, P., Sundararajan, N.: A fast and accurate on-line sequential learning algorithm for feedforward networks. IEEE Transactions on Neural Networks 17(6), 1411–1423 (2006)
17. Cristianini, N., Taylor, J.S.: An Introduction to Support Vector Machines. Cambridge University Press, Cambridge (2000)
18. Suresh, S., Dong, K., Kim, H.J.: A sequential learning algorithm for self-adaptive resource allocation network classifier. Neurocomputing 73(16-18), 3012–3019 (2010)
19. Suresh, S., Venkatesh Babu, R., Kim, H.J.: No-reference image quality assessment using modified extreme learning machine classifier. Applied Soft Computing 9(2), 541–552 (2009)
20. Suresh, S., Sundararajan, N., Saratchandran, P.: A sequential multi-category classifier using radial basis function networks. Neurocomputing 71(7-9), 1345–1358 (2008)

Urban Stormwater Runoff Prediction Using Recurrent Neural Networks

Nian Zhang

University of the District of Columbia,
Department of Electrical and Computer Engineering,
4200 Connecticut Avenue, NW, Washington D.C. 20008 USA
nzhang@udc.edu

Abstract. It has been recognized that urban stormwater pollution can be a large contributor to the water quality problems of many receiving waters. Stormwater pollution is one of most important issues the District of Columbia faces. The downtown core of the District is serviced by combined sewer system. Therefore, evaluations of stormwater runoff are necessary to enhance the performance of an assessment operation and develop better water resources management and plan. In order to accomplish the goal, a predictive model based on recurrent neural networks with the Levenberg-Marquardt backpropagation training algorithm is developed to forecast the stormwater runoff using the precipitation and the previous stormwater runoff. This computational modeling tool explored a new computational intelligence solution for monitoring and controlling urban water pollution in the District of Columbia. The experimental results show that Levenberg-Marquardt backpropagation training algorithm proved to be successful in training the recurrent neural network for the stormwater runoff prediction.

Keywords: Runoff Quantity and Quality Prediction, Recurrent Neural Networks, Levenberg-Marquardt Backpropagation Training Algorithm.

1 Introduction

Stormwater runoff is unfiltered water that reaches streams, lakes, sounds, and oceans by means of flowing across impervious surfaces. These surfaces include roads, parking lots, driveways, and roofs. It has been recognized that urban stormwater pollution can be a large contributor to the water quality problems of many receiving waters, as runoff transports a wide spectrum of pollutants to local receiving waters and their cumulative magnitude is large [1][2]. The discharge of untreated storm water and combined sewer overflows (CSOs) constitutes a source of pollution to the receiving waters of the District of Columbia as well as the Chesapeake Bay.

Forecasting of runoff quantity and quality benefit substantially from the progress of computational intelligence techniques, particularly neural networks. Computational intelligence relies on heuristic algorithms such as in fuzzy systems, neural networks, and evolutionary computation [3]. In addition, it also embraces techniques that use swarm intelligence, chaos theory, artificial immune systems, and wavelets. Comparatively, various runoff forecast models based on neural networks perform much better in

© Springer-Verlag Berlin Heidelberg 2011

accuracy than many conventional prediction models [4]-[10]. However, a fact could not be neglected that most of such existing neural networks based models have not yet satisfied researchers and engineers in forecast precision so far, and the generalization capability of these networks needs further improving. For example, most publications used the feedforward neural networks with Backpropagation algorithms. However, a critical "drawback" of the Backpropagation algorithm is the local minima problem caused by neuron saturation in the hidden layer [11]. Because of this, the algorithm cannot converge to the minimum error, and thus it cannot get accurate prediction results.

To overcome the above challenges, it is extremely important to investigate new models with the potential for higher rates of prediction. According to the time series prediction competition results in the 2006, 2008, and 2010 Artificial Neural Network & Computational Intelligence Forecasting Competitions [12][13], recurrent neural networks, wavelet neural networks, particle swarm optimization methods, and fuzzy neural networks etc. have been widely recognized as the best models for time series prediction [14]-[27]. Because time series prediction is a generalized form of runoff prediction, we can expect these models will also work the best for the specific runoff prediction. However, these prospective methods have never been used for the runoff quantity [4-10] and quality prediction problems [28][29][30]. Therefore, this paper is intended to propose a recurrent neural network based model with the Levenberg-Marquardt backpropagation training algorithm to forecast the stormwater runoff in terms of the precipitation and the previous stormwater runoff.

This paper is organized as follows: In Section 2, the data and methods are introduced. Study area and stormwater runoff data will be explained. Network architecture and network learning algorithm are presented. In Section 3, experimental results including the error autocorrelation function, input-error cross-correlation function, and time series response are demonstrated. In Section 4, the conclusions are given.

2 Data and Methods

2.1 Study Area and Stormwater Runoff Data

Since the downtown core of the District of Columbia has been deteriorated with increasing levels of stormwater pollution, we thus pay particular attention to the stormwater runoff data within the District of Columbia.

Rea-time stormwater runoff water data are obtained from the U.S. Geological Survey (USGS)'s national water information system. Fig. 1 shows all the water stations within or near Washington D.C. The dots stand for the water stations. The big square in the middle represents the boarder of the District of Columbia. The circle on the bottom of the square encloses the Four Mile Run water station. The Four Mile Run stream station at Alexandria, VA is selected to retrieve data because a) it is located within the District, and b) it is the only site that provides both the precipitation data and discharge data within the District. Both datasets will be used to predict the stormwater runoff discharge.

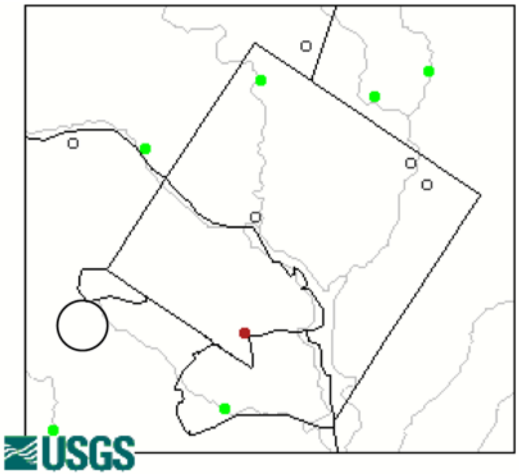


Fig. 1. Streamflow sites within or near the Washington D.C. The dots on this map depict streamflow sites. The big square in the middle represents the boarder of the District of Columbia. The circle on the bottom of the square indicates the Four Mile Run water station.

The input datasets are precipitation (inches) and discharges (cubic feet per second) during July 31, 2010 to November 20, 2010, as shown in Fig. 2, and Fig. 3, respectively.

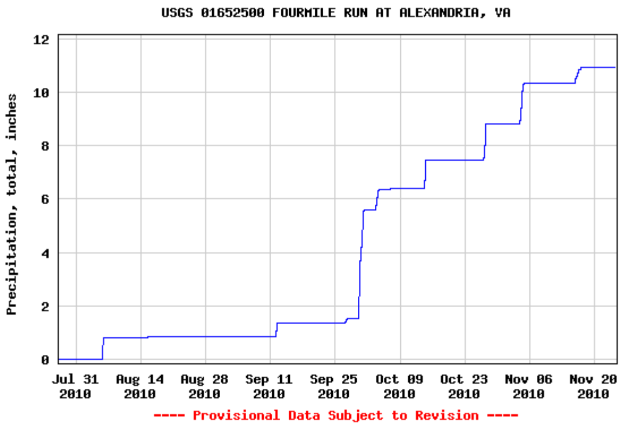


Fig. 2. Precipitation data (inches) collected at the Four Mile Run site at Alexandria, VA during July 31, 2010 to November 20, 2010

Real-time data typically are recorded at 15- to 60-minute intervals. Therefore, each figure plots 34721 data during the 120 days. Input 'precipitation' is a 34721x1 matrix, representing dynamic data: 34721 timesteps of 1 element. Target 'discharge' is a 34721x1 matrix, representing dynamic data: 34721 timesteps of 1 element. The target data are randomly divided up into 34721 timesteps. 70% of the data is used for training, which contributes 24305 target timesteps. They are presented to the network during training, and the network is adjusted according to its error. 15% of the data is used for validation, which accounts for 5208 target timesteps. They are used to measure network generalization, and to halt training when generalization stops improving. The last 15% of the data is used for testing, which has 5208 target timesteps. They provide an independent measure of network performance during and after training.

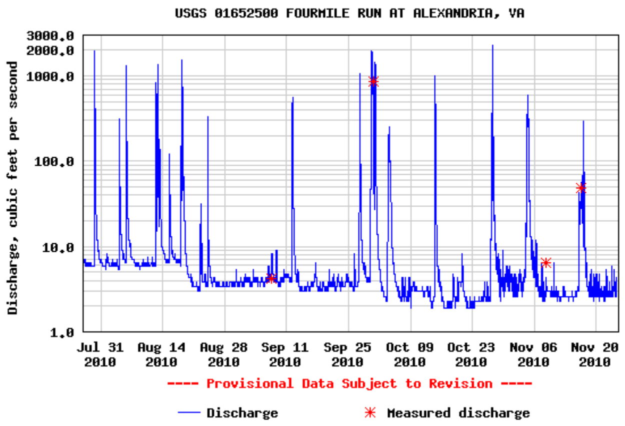


Fig. 3. Discharge data (cubic feet per second) collected at the Four Mile Run site at Alexandria, VA during July 31, 2010 to November 20, 2010

2.2 Network Architecture

A predictive model is to be developed to predict future values of runoff discharge, based on the precipitation and previous runoff discharge. Such a model can be represented mathematically by predicting future values of the discharges time series y(t) from past values of that time series and past values of the precipitation time series x(t). This form of prediction can be written as follows:

$$y(t) = f(y(t-1), \ldots, y(t-d), x(t-1), \ldots, x(t-d)) \tag{1}$$

The network architecture is shown in Fig. 4. The network is a two-layer feedforward network, with a sigmoid transfer function in the hidden layer and a linear transfer function in the output layer. This network also uses tapped delay lines to store previous values of the x(t) and y(t) sequences. Note that the output of the network, y(t), is fed back to the input of the network through delays, since y(t) is a function of y(t-1), y(t-2), ..., y(t-d). However, the network will be created and trained in open loop form as shown in Fig. 4. Open loop (single-step) training is more efficient than closed loop (multi-step) training. Open loop allows us to supply the network with correct past outputs as we train it to produce the correct current outputs. After training, the network may be converted to closed loop form, or any other form, that the application requires. Because the true output is available during the training of the network, we can use the open-loop architecture shown in Fig. 4, in which the true output is used instead of feeding back the estimated output. This has two advantages. The first is that the input to the feedforward network is more accurate. The second is that the resulting network has a purely feedforward architecture, and therefore a more efficient algorithm can be used for training.

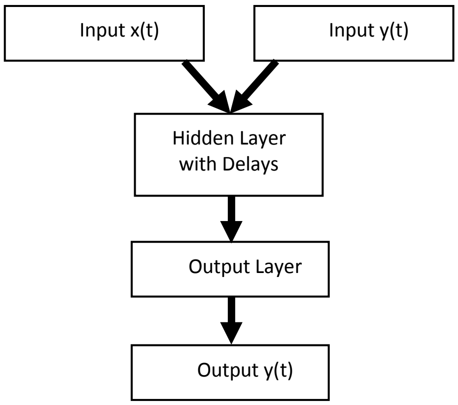


Fig. 4. Network architecture. The network is a two-layer feedforward network, with a sigmoid transfer function in the hidden layer and a linear transfer function in the output layer. This network also uses tapped delay lines to store previous values of the x(t) and y(t) sequences.

2.3 Network Learning Algorithm

The network is trained using Levenberg-Marquardt backpropagation training algorithm because it can achieve better accuracy and convergence speed than the gradient descent and conjugate gradient training algorithms. Training automatically stops when generalization stops improving, as indicated by an increase in the mean square error of the validation samples.

Levenberg-Marquardt backpropagation algorithm applies a function minimization routine, which can back propagate error into the network layers as a means of improving the calculated output. It can be represented as [31]:

$$\Delta x_k = x_{k+1} - x_k = \alpha_k p_k \tag{2}$$

where x_k is the current estimation point for a function $G(x)$ to be minimized at the kth stage, p_k is the search vector and α_k is the learning rate, a scalar quantity greater than zero. The learning quantity identifies the step size for each repetition along p_k . Computation of p_k will depend on the selected learning algorithm. Equation (2) can be rewritten as:

$$x_{k+1} = x_k - A_k^{-1} g_k \tag{3}$$

where g_k and A_k are the first and the second derivative of $G(x)$ with respect to x, and A_k^{-1} is the inverse function of A_k. Equation (3) can be further derived as:

$$x_{k+1} = x_k - \frac{1}{2\mu_k} \nabla G(x) \tag{4}$$

where $\nabla G(x)$ is the gradient of $G(x)$.

In this equation, if the value of coefficient μ_k is decreased to zero the algorithm becomes Gauss-Newton. The algorithm begins with a small value for μ_k. If a step does not yield a smaller value for G(x), the step is repeated with μ_k multiplied by a factor greater than one. Eventually, $G(x)$ will decrease since it would take a small step in the direction of the steepest descent. If a step does produce a smaller value for $G(x)$, then it is divided by the specified factor for the next step, so that the algorithm will approach Gauss-Newton, which should provide faster convergence. The algorithm provides a neat compromise between the speed of Newton's method and the guaranteed convergence of steepest descent.

3 Experimental Results

The number of neurons in the hidden layer and the number of delays in the tapped delay lines have been tried plenty of times until the network has performed well after training.

The best number of hidden neurons is 50, and the best number of delays in the tapped delay lines is 11. Both of these numbers are the maximum number that we can set; otherwise the network will be out of memory.

3.1 Error Autocorrelation Function

The error autocorrelation function is used to validate the network performance. Fig. 5 displays the error autocorrelation function. It describes how the prediction errors are related in time. For a perfect prediction model, there should only be one nonzero value of the autocorrelation function, and it should occur at zero lag. (This is the mean square error.) This would mean that the prediction errors were completely uncorrelated with each other (white noise). If there was significant correlation in the prediction errors, then we can improve the prediction accuracy by increasing the number of delays in the tapped delay lines. In this case, the correlations, except for the one at zero lag, fall approximately within the 95% confidence limits around zero, so the model seems to be adequate.

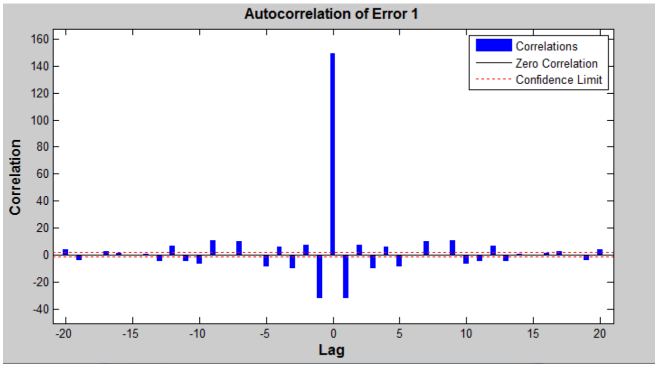


Fig. 5. Error autocorrelation function. It describes how the prediction errors are related in time.

3.2 Input-Error Cross-Correlation Function

This input-error cross-correlation function illustrates how the errors are correlated with the input sequence $x(t)$. For a perfect prediction model, all of the correlations should be zero. If the input is correlated with the error, then we can improve the prediction accuracy by increasing the number of delays in the tapped delay lines. In this case, all of the correlations fall within the confidence bounds around zero, as shown in Fig. 6.

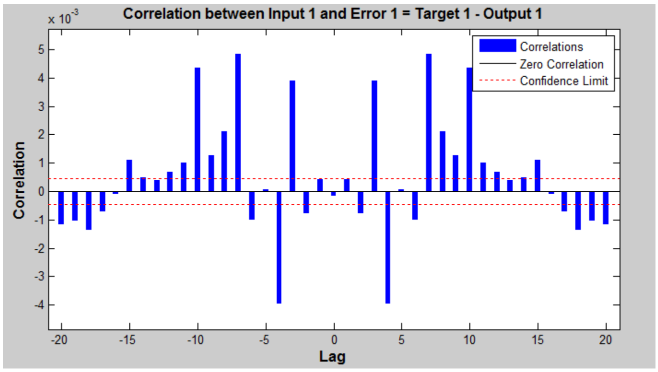


Fig. 6. The input-error cross-correlation function. It describes how the prediction errors are related in time.

3.3 Time Series Response

Fig. 7 demonstrates the time series response. The top plot displays the outputs and targets versus time. For each selected time point for training, testing and validation, all the training targets, training outputs, validation targets, validation outputs, test targets, and test outputs are plotted. The bottom plot shows the error versus time. At those selected time point for training, testing and validation, the errors for training target, validation target, and test target are plotted. The solid line is used to measure the magnitude of errors. As can be seen from Fig. 7, the prediction of training data, validation data, and test data are all satisfactory.

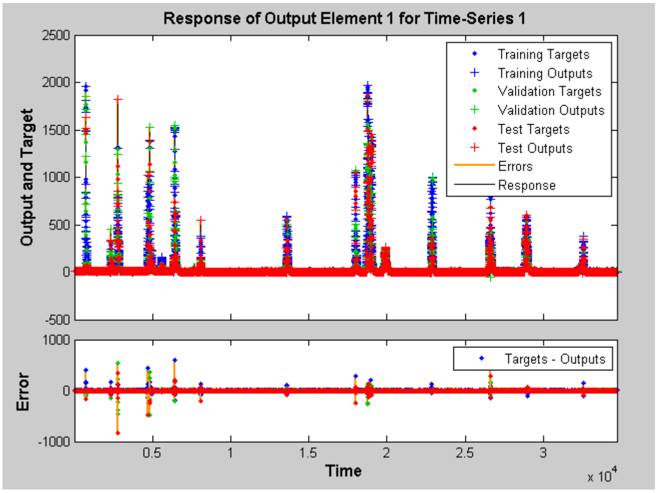


Fig. 7. Error autocorrelation function. It describes how the prediction errors are related in time.

4 Conclusions

This paper provides a predictive model based on recurrent neural networks with the Levenberg-Marquardt backpropagation training algorithm to forecast the stormwater runoff using the precipitation and the previous stormwater runoff.

In this research, the stormwater runoff at the Four Mile Run stream station was studied, because of its impact to the District of Columbia and Potomac River. The input data are precipitation and discharges with 120 days duration from July 31, 2010 to November 20, 2010. The Levenberg-Marquardt backpropagation training algorithm is used because of its high accuracy and convergence speed. The error autocorrelation function and input-error cross-correlation function were used to validate the network performance. The number of neurons in the hidden layer and the number of delays in the tapped delay lines have been tried plenty of times until the network has performed well after training. The best number of hidden neurons is 50, and the best number of delays in the tapped delay lines is 11. The experimental results show that Levenberg-Marquardt backpropagation training algorithm proved to be successful in training the recurrent neural network for the stormwater runoff prediction.

Acknowledgment

The author would like to thank the US National Science Foundation (Award #: OISE-1066140).

References

1. U.S. Environmental Protection Agency (EPA). Washington, DC, Protecting Water Quality from Urban Runoff. Document No. EPA 841-F-03-003 (2003)
2. United States. National Research Council. Washington, DC, Urban Stormwater Management in the United States, pp. 18–20 (2008)
3. Haykins, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall, Englewood Cliffs (1998)
4. Bai, P., Song, X., Wang, J., Shi, W., Wang, Q.: A Hillslope Infiltration and Runoff Prediction Model of Neural Networks Optimized by Genetic Algorithm. In: Brennan, R., Fleck II, J., van der Meer, S. (eds.) MACE 2010. LNCS, vol. 6473, pp. 1256–1259. Springer, Heidelberg (2010)
5. Hui, S., Xinxia, L.: Multi-scale RBF Prediction Model of Runoff Based on EMD Method. In: 2010 Third International Conference on Information and Computing (ICIC), pp. 296–299 (2010)
6. Guo, W., Wang, H., Xu, J., Zhang, Y.: RBF Neural Network Model Based on Improved PSO for Predicting River Runoff. In: 2010 International Conference on Intelligent Computation Technology and Automation (ICICTA), pp. 968–971 (2010)
7. Bo, H., Dong, X., Deng, X.: Application of GA-ANN Hybrid Algorithms in Runoff Prediction. In: 2010 International Conference on Electrical and Control Engineering (ICECE), pp. 5039–5042 (2010)
8. Guo, J., Xiong, W., Chen, H.: Application of Rough Set Theory to Multi-factor Medium and Long-period Runoff Prediction in Danjing Kou Reservoir. In: The Sixth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp. 177–182 (2009)
9. Zhang, J., Zhu, C.: Prediction of Reservior Runoff Using RBF Neural Network-Grey System United Model. In: IITA International Conference on Control, Automation and Systems Engineering (CASE), pp. 43–46 (2009)
10. Solaimani, K.: Rainfall-runoff Prediction Based on Artificial Neural Network (A Case Study: Jarahi Watershed). American-Eurasian J. Agric. & Environ. Sci. 5(6), 856–865 (2009)
11. Bi, W., Wang, X.: Avoiding the Local Minima Problem in Backpropagation Algorithm with Modified Error Function. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E88-A(12) (2005)
12. Artificial Neural Network & Computational Intelligence Forecasting Competition in (2006, 2008, 2010), `http://www.neural-forecasting-competition.com/`
13. Lendasse, Oja, E., Simula, O., Verleysen, M.: Time Series Prediction Competition: The CATS Benchmark. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, July 25-29, pp. 1615–1620 (2004)
14. Cai, X., Zhang, N., Venayagamoorthy, G.K., Wunsch II, D.C.: Time Series Prediction with Recurrent Neural Networks Trained by a Hybrid PSO-EA Algorithm. Neurocomputing 70(13-15), 2342–2353 (2007)
15. Cai, X., Zhang, N., Venayagamoorthy, G.K., Wunsch II, D.C.: Time Series Prediction with Recurrent Neural Networks Using Hybrid PSO-EA Algorithm. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, July 25-28, vol. 2, pp. 1647–1652 (2004)
16. Sarkka, S., Vehtari, A., Lampinen, J.: Time Series Prediction by Kalman Smoother with Cross Validated Noise Density. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)

17. Kurogi, S., Ueno, T., Sawa, M.: Batch Learning Competitive Associative Net and Its Application to Time Series Prediction. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
18. Palacios-Gonzalez, F.: A SVCA Model for the Competition on Artificial Time Series. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
19. Herrera Maldonado, L.J., Pomares, H., Rojas, I., Gonzalez, J., Awad, M.: MultiGrid-Based Fuzzy Systems for Time Series Forecasting: CATS Benchmark IJCNN Competition. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
20. Simon, G., Lee, J.A., Verleysen, M., Cottrell, M.: Double Quantization Forecasting Method for Filling Missing Data in the CATS Time Series. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
21. Verdes, P.F., Granitto, P.M., Szeliga, M.I., Rebola, A., Ceccatto, H.A.: Prediction of the CATS benchmark exploiting time-reversal symmetry. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
22. Wichard, J., Ogorzalek, M.: Time Series Prediction with Ensemble Models. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
23. Kozma, R., Beliaev, I.: Time Series Prediction Using Chaotic Neural Networks: Case Study of IJCNN CATS Benchmark Test. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
24. Kong, S.G.: Time Series Prediction with Evolvable Block-based Neural Networks. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
25. Cellier, F.E., Nebot, A.: Multi-resolution Time-Series Prediction Using Fuzzy Inductive Reasoning. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
26. Crone, S.F., Kausch, H., Pressmar, D.: Prediction of the CATS benchmark using a Business Forecasting Approach to Mulitlayer Perceptron Modelling. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
27. Eleuteri, F., Acernese, F., Barone, F., De Rosa, R., Milano, L.: A Hierarchical Bayesian Learning Scheme for Autoregressive Neural Networks: Application to the CATS Benchmark. In: INNS-IEEE International Joint Conference on Neural Networks (IJCNN), Budapest (2004)
28. Najah, A., Elshafie, A., Karim, O., Jaffar, O.: Prediction of Johor River Water Quality Parameters Using Artificial Neural Networks. European Journal of Scientific Research 28(3), 422–435 (2009)
29. Wang, X., Lv, J., Xie, D.: A Hybrid Approach of Support Vector Machine with Particle Swarm Optimization for Water Quality Prediction. In: The 5th International Conference on Computer Science and Education (ICCSE), pp. 1158–1163 (2010)
30. Miao, Q., Zhao, L., Gao, A., Liu, C., Miao, S.: Optimization Design and Application of Water Quality Evaluation Model based on BP Neural Network. In: The 4th International Conference on Bioinformatics and Biomedical Engineering, pp. 1–5 (2010)
31. Hagan, M.T., Demuth, H.B., Beale, M.: Neural Network Design, 2nd edn. PWS Publishing Company, Boston (1996)

Maximal-Margin Approach for Cost-Sensitive Learning Based on Scaled Convex Hull

Zhenbing Liu

School of Electronic Engineering and Automation,
Guilin University of Electronic Technology,
541004, Guilin, China
liuzb0618@hotmail.com

Abstract. In this paper, a new maximal margin method, scaled convex hull (SCH) method is proposed to solve the cost-sensitive learning. By providing different SCH with a different scale factor, the initial overlapping SCHs can be reduced to become separable, and the existing methods can be used to find the separating hyperplane. The new method changes the distribution of the sample, which assigns different scale factor. The experiment results are used to validate the effectiveness of the scaled convex hull and its simplicity.

Keywords: scaled convex hull, cost-sensitive, maximal margin.

1 Introduction

In classical machine learning or data mining settings, the classifiers usually try to minimize the number of errors which they will make in dealing with new data. Such a setting is valid only when the costs of different errors are equal. Unfortunately, the costs of different errors are often unequal in many real-world applications. For example, in medical diagnosis, the cost of erroneously diagnosing a patient to be healthy may be much bigger than that of mistakenly diagnosing a healthy person as being sick, because the former kind of error may result in the loss of a life.

In fact, cost-sensitive learning has already attracted much attention from the machine learning and data mining communities. As it has been stated in the Technological Roadmap of the MLnetII project (European Network of Excellence in Machine Learning), the inclusion of costs into learning has been regarded as one of the most relevant topics of future machine learning research. During the past years, many cost-sensitive learning methods have been developed [1] [2] [3] [4] [5]. However, although there are much research efforts devoted to making decision trees cost-sensitive [6] [7] [8], and some studies discuss cost-sensitive neural networks [9] [10], while it is usually not feasible to apply cost-sensitive decision tree learning methods to neural networks directly. For example, the instance-weighting method requires the learning algorithm accept weighted-examples, which is not a problem for C4.5 decision trees but is difficult for common feed forward neural networks.

In this paper, by introducing the notion of scaled convex hull (SCH), a new maximal margin method is proposed to solve the cost-sensitive learning. By providing different SCH with a different scale factor, the initial overlapping SCHs can be

© Springer-Verlag Berlin Heidelberg 2011

reduced to become separable. Once they are separable, we can find the nearest point pair between them using the existing algorithms, and the separating hyperplane a) bisects, and b) is normal to the line segment joining these two nearest points. This separating hyperplane obtains the maximal margin between SCHs, resulting in a maximal-margin classifier. This viewpoint is the same as the reduced convex hull (RCH) framework for SVM classifiers, so it can be seen as a variant of SVMs. It simplifies the computation of nearest point pair between the SCHs, which is independent to the reduction factor. Besides, the SCH has the same shape as the original convex hull when the scale factor changes, so we call it scaled convex hull. The new method changes the distribution of the sample, which assigns different scale factor. The experiment results are used to validate the effectiveness of the scaled convex hull and its simplicity.

2 Scaled Convex Hull

Given the training data, the SVM training algorithm obtains the optimal separating hyperplane between two classes of training samples: $f(x) = <w, x> + t$, where w is the weight vector and t is the bias. The optimal separating hyperplane maximizes the margin, i.e. $2/\|w\|$ (or, alternatively, minimizes $\|w\|^2/2$) [11]. This classification task, expressed in its dual form, is equivalent to finding the pair of nearest points between the convex hulls (each is generated by the training patterns of each class), and the maximal margin hyperplane a) bisects, and b) is normal to the line segment joining these two nearest points [12]. A convex hull generated by training patterns of one class $X = \{x_i, x_i \in \mathrm{R}^d, i = 1, 2, \cdots, k\}$ is defined as

$$conv(\mathrm{X}) = \left\{ \sum_{i=1}^{k} a_i x_i : \sum_{i=1}^{k} a_i = 1, x_i \in \mathrm{X}, 0 \le a_i \le 1 \right\} \tag{1}$$

For nonseparable problems, i.e., the convex hulls of the patterns in the feature space are overlapping. The framework of SCH is introduced to transform them to separable ones.

The scaled convex hull (SCH) of the set $\mathrm{X} = \{x_i, x_i \in \mathrm{R}^d, i = 1, 2, \cdots, k\}$ with the non-negative reduction factor $0 \le \lambda \le 1$, denoted as $\mathrm{S}(\mathrm{X}, \lambda)$, is defined as

$$\mathrm{S}(\mathrm{X},\lambda) = \left\{ \lambda \sum_{i=1}^{k} a_i x_i + (1-\lambda)m : \sum_{i=1}^{k} a_i = 1, x_i \in \mathrm{X}, 0 \le a_i \le 1, m = \frac{1}{k}\sum_{i=1}^{k} x_i \right\} \tag{2}$$

It can also be rewritten as

$$\mathrm{S}(\mathrm{X},\mu) = \left\{ \sum_{i=1}^{k} a_i (\lambda x_i + (1-\lambda)m) : \sum_{i=1}^{k} a_i = 1, x_i \in \mathrm{X}, 0 \le a_i \le 1, m = \frac{1}{k}\sum_{i=1}^{k} x_i \right\} \tag{3}$$

Where m, the mean value of all original points, is called the centroid point of the X.

For a given λ, every point $\lambda \sum_i a_i x_i + (1-\lambda)m$ of $\mathrm{S}(\mathrm{X}, \mu)$ is the linear combination of the centroid m and the point $\sum_{i=1}^{k} a_i x_i$ of the original convex hull, i.e., the point $\lambda \sum_i a_i x_i + (1-\lambda)m$ of the SCH lies on the line segment connecting $\sum_{i=1}^{k} a_i x_i$ and the centroid m . In fact, the ratio of the distance between $\lambda\sum_i a_i x_i + (1-\lambda)m$ and the centroid m to the distance between $\sum_{i=1}^k a_i x_i$ and the centroid m is the constant λ. So the shape of the SCH $\mathsf{S}(\mathsf{X},\mu)$ is the same as that of the original convex hull $conv(\mathsf{X})$ (This is why we call it scaled convex hull). It seems that $\mathsf{S}(\mathsf{X},\mu)$ is obtained by "reducing" $conv(\mathsf{X})$ towards the centroid m by λ which controls the size of the SCH. Furthermore, the reduction factor λ can be set different for each class, reflecting the importance of each class.

For convenience, we denote the "reduced" point $\lambda x_i + (1-\lambda)m$ as x_i' and $\mathsf{X}' = \{x_i', i = 1, 2, \cdots, k\}$. Then, the SCH can be rewritten as

$$\mathsf{S}(\mathsf{X},\mu) = \left\{ \sum_{i=1}^k a_i x_i' : \sum_{i=1}^k a_i = 1, x_i \in \mathsf{X}, 0 \le a_i \le 1 \right\} \tag{4}$$

From (4), the SCH can be seemed as a convex hull generated by the points x_i' s of the "reduced" set X' , so it can also be denoted as $conv(\mathsf{X}')$. Thus, the candidate extreme point set of $\mathsf{S}(\mathsf{X},\mu)$ is X' , having the same number of elements as the original X when $\lambda \ne 0$.

In this way, the initial overlapping convex hulls can be reduced to become separable by a suitable selection of λ, see Fig. 1 .

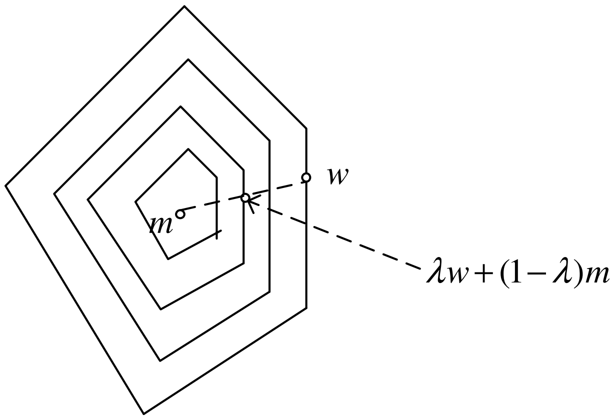


Fig. 1. Geometric interpretation of SCH

In the sequel, we will prove some theorems and propositions useful to the SCH notion and form the basis for the development of the novel algorithm which is proposed in this paper.

Proposition 1. when $\lambda = 1$, the SCH $\mathsf{S}(\mathsf{X},\mu)$ is the original convex hull; and when $\lambda = 0$ it becomes the centroid.

Proof: substituting $\lambda = 1$ and $\lambda = 1$ into (3) respectively, we can get the result.

Proposition 2. The SCH and the original convex hull have the same centroid.

Proof: From (3), we know that the SCH can be seemed as spanned by the points $x_i^{'}$ which correspond to the point x_i for $i = 1, 2, \cdots, k$. So the centroid of the SCH is

$$\begin{aligned}\sum_i x_i^{'} / k &= \sum_i (\lambda x_i + (1-\lambda) m / k) \\ &= \sum_i \lambda x_i / k - \lambda m + m \\ &= m \ ,\end{aligned}$$

i.e., the SCH and the original convex hull have the same centroid.

The SCH $\mathrm{S}(\mathrm{X}, \mu)$ can be rewritten as

$$\mathrm{S}(\mathrm{X}, \lambda) = \left\{ \sum_{i=1}^{k} b_i x_i, \sum_{i=1}^{k} b_i = 1, x_i \in \mathrm{X}, \frac{1-\lambda}{k} \le b_i \le \lambda + \frac{1-\lambda}{k} \right\} \tag{5}$$

3 SCH for Cost-Sensitive

Suppose $\mathrm{X} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ is the training set for a binary classification problem where x_k is the training pattern belonging to one of two classes and $y_k \in \{1, -1\}$ is its corresponding class label. In the following theorem, we will give the relation between the classical C-SVM with slack variables and SCHs, which opens the road for applying nearest point algorithms for the nonseparable problems. This viewpoint is the same as the SVM framework. Before giving the theorem, we first define some notations that have the same meaning in the following sections.

$$\mathrm{I}^+ = \{i : y_i = 1\},\ \mathrm{I}^- = \{i : y_i = -1\},\ \mathrm{I} = \mathrm{I}^+ \cup \mathrm{I}^-$$

$$\mathrm{X}^+ = \{x_i, i \in \mathrm{I}^+\},\ \mathrm{X}^- = \{x_i, i \in \mathrm{I}^-\}$$

$$m^+ = \sum_{i \in I^+} x_i / n^+ \text{, where } n^+ = |\mathrm{I}^+|$$

$$m^- = \sum_{i \in I^-} x_i / n^- \text{, where } n^- = |\mathrm{I}^-|$$

$$\mathrm{X}^{+'} = \left\{ x_i^{'} = \lambda^+ x_i + (1-\lambda^+) m^+, i \in \mathrm{I}^+ \right\}$$

$$\mathrm{X}^{-'} = \left\{ x_i^{'} = \lambda^- x_i + (1-\lambda^-) m^-, i \in \mathrm{I}^- \right\}$$

$$\mathrm{S}(\mathrm{X}^+, \lambda^+) = \{ \sum_i a_i (\lambda^+ x_i + (1-\lambda^+) m^+), i \in \mathrm{I}^+ \}$$

$$\mathrm{S}(\mathrm{X}^-, \lambda^-) = \{ \sum_i a_i (\lambda^- x_i + (1-\lambda^-) m^-), i \in \mathrm{I}^- \}$$

By providing different SCH with a different scale factor, the cost-sensitive can be addressed, i.e., providing the positive SCH with bigger scale factor and the two SCHs will have the same area, and the resulting classifier will misclassify less positive points.

By a suitable selection of the reduction factors, the initial overlapping convex hulls can be reduced to become separable, and the maximal margin hyperplane a) bisects, and b) is normal to the line segment joining these two nearest points, as shown in Fig. 2.

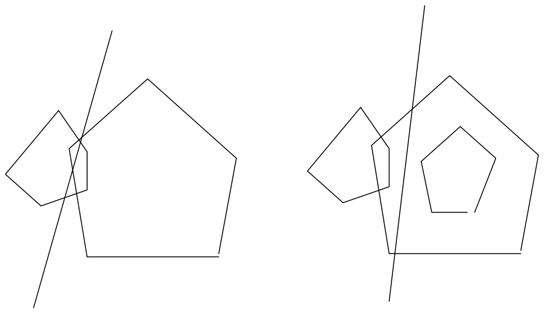


Fig. 2. The SCH for cost-sensitive

Theorem: The classifier associated with the SCHs converges to that of the cost-sensitive C-SVM by a suitable selection of parameters, as the number of training samples tends to infinity.

Proof: The problem of finding the pair of closest point in the separable SCHs can be written as the following optimization problem

$$\min_b \left\| \sum_{i\in I^+} b_i x_i - \sum_{i\in I^-} b_i x_i \right\| / 2 \tag{6}$$

subject to

$$\sum_{i\in I^+} b_i = 1, \sum_{i\in I^-} b_i = 1$$
$$(1-\lambda^+)/n^+ \le b_i \le \lambda^+ + (1-\lambda^+)/n^+, \text{ for } i \in \mathrm{I}^+$$
$$(1-\lambda^-)/n^- \le b_i \le \lambda^- + (1-\lambda^-)/n^-, \text{ for } i \in \mathrm{I}^-$$

By rescaling the objective function and using the class labels, we can rewrite this as

$$\min_b \sum_{i,j\in I} b_i b_j y_i y_j < x_i, x_j > /4 \tag{7}$$

subject to

$$\sum_{i\in I} b_i y_i = 0, \sum_{i\in I} b_i = 2$$
$$(1-\lambda^+)/n^+ \le b_i \le \lambda^+ + (1-\lambda^+)/n^+, \text{ for } i \in I^+,$$
$$(1-\lambda^-)/n^- \le b_i \le \lambda^- + (1-\lambda^-)/n^-, \text{ for } i \in I^-.$$

The C-SVM approach with margin slack variables for nonseparable problems is

$$\max_{w,b} \|w\|^2 / 2 + C^+ \sum_{i\in I^+} \xi_i + C^- \sum_{i\in I^-} \xi_i ;$$
$$y_i(w \cdot x_i) + b \ge 1 - \xi_i,\ i = 1, 2, \cdots, n ; \tag{8}$$
$$\xi_i > 0 。$$

Where C is a positive constant.

The dual is

$$\begin{aligned} &\min_{c} \frac{1}{4} \sum_{i,j\in I} a_i a_j y_i y_j (x_i \cdot x_j) \;; \\ &\text{s.t. } \sum_{i\in I} a_i y_i = 0, \;\; \sum_{i\in I} a_i = 2 \;; \\ &0 \le a_i \le C^+, i \in I^+ \;; \\ &0 \le a_i \le C^-, i \in I^- \circ \end{aligned} \tag{9}$$

The only difference between the two optimization problems (7) and (9) is that the coefficients of the latter are upper-bounded by a constant C, and the coefficients of the former are lower-bounded and upper-bounded by some constants. When n^+ and n^- tend to infinity, $(1-\lambda^+)/n^+$ and $(1-\lambda^-)/n^-$ converge to zero, i.e., the constraints (7) converge to (9), hence the solution of (7) converges to the solution of (9).

This theorem proves that the separating hyperplane associated with the SCHs converges to the Classic C-SVM separating hyperplane, so the method can be regarded as an alternative to construct SVM classifiers.

4 The MDM Algorithm for SCH Classifier

The so-called MDM algorithm for solving the linearly separable SVM problem has been presented recently in [13]. One important advantage of the algorithm is that, the adaptation only involves points of training set that may be an extreme point of the original convex hulls. Since our two SCHs, $\mathsf{S}(\mathsf{X}^+, \lambda^+)$ and $\mathsf{S}(\mathsf{X}^-, \lambda^-)$, are spanned by the set $\mathsf{X}^{+'}$ and $\mathsf{X}^{-'}$ containing the extreme points of each SCH respectively, the algorithm is easily generalized to find an ε-optimal separating hyperplane for them once the two SCHs are separable.

First, two definitions are given.

Definition 1. let Q be a convex and compact subset of R^d, $\eta(x) = \max_{x\in Q} < x, y >$ is called the support function of Q.

Definition 2. $s(y)$ is called a contact function If $s(y) \in \{x :< x, y >= \eta(x)\} \cap \mathsf{Q}$, $y \neq 0$ and $s(y) \in \mathsf{Q}$.

Then, the MDM algorithm for finding the ε-optimal separating hyperplane can be described as follows:

Step 1: set the vector $w_1 = \sum_{i\in I^+} a_i x_i^{'}$ and $w_2 = \sum_{i\in I^-} a_i x_i^{'}$. Set the stop criterion ε.

Step 2: find the vector $x_t^{'}$ closest to the hyperplane as $t = \arg\min m(x_i^{'})$, where $m(x_i^{'}) =< x_i^{'} - w_2, w_1 - w_2 > / \|w_1 - w_2\|$ for $i \in \mathrm{I}^+$ and $m(x_i^{'}) =< x_i^{'} - w_1, w_2 - w_1 > / \|w_1 - w_2\|$ for $i \in \mathrm{I}^-$.

If the ε -optimal condition $\|w_1 - w_2\| - m(x_t^{'}) < \varepsilon$ holds, then the vector $w = w_1 - w_2$ and $b = (\|w_1\|^2 + \|w_2\|^2)/2$ gives the ε -solution; otherwise, let $z = w_1 - w_2$ and go to step 3.

Step 3: if $x_t^{'} \in X^{+'}$, find an index $t_{\min} \in I^+$ such that $< -z, x_{t_{\min}}^{'} >= \min\{< -z, x_i^{'} >: a_i > 0, i \in I^+\}$. Let $d = s(-z) - x_{t_{\min}}^{'}$, $\bar{z} = z + a_{t_{\min}} * d$. Let z^{new} be the point of minimum norm on the line segment joining z and $\bar{z}$. Then let $w_2^{new} = w_2$, $w_1^{new} = w_1 + w_2^{new}$.

Otherwise, find an index $t_{\min} \in I^-$ such that $< -z, x_{t_{\min}}^{'} >= \min\{< -z, x_i^{'} >: a_i > 0, i \in I^-\}$. Let $d = s(-z) - x_{t_{\min}}^{'}$, $\bar{z} = z + a_{t_{\min}} * d$.

Then let $w_1^{new} = w_1$, $w_2^{new} = w_2 + w_1^{new}$. Continue with Step 2.

5 Experiments

In order to extensively investigate the performance (the average cost) of the new algorithm presented here, two available test datasets from UCI machine learning Repository (Heart and German) have been used. The cost-sensitive SVM and the proposed algorithm were implemented, tested and compared. Each algorithm was trained and tested for each dataset, under the RBF kernel in order to achieve the same accuracy referred in the literature [14]. The same validation method and the same data realizations are used. The results of the runs are showed in Fig. 3 and Fig. 4.

It can be seen that the proposed method gets less cost and is better than the SVM.

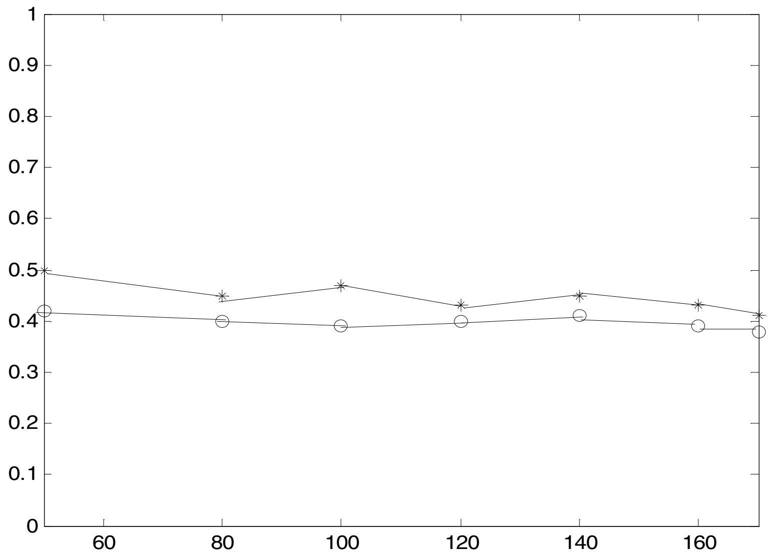


Fig. 3. The average cost of SVM("o") and the proposed method ("*") for the Heart

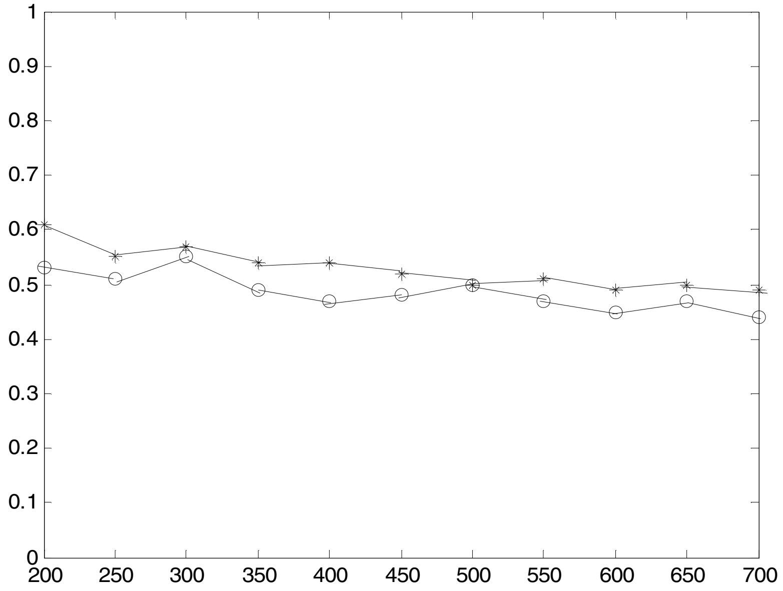


Fig. 4. The average cost of SVM("o") and the proposed method ("*") for the German

6 Conclusions

In this paper, by introducing the notion of scaled convex hull (SCH), a new maximal margin method is proposed to solve the cost-sensitive learning. The new method changes the distribution of the sample, which assigns different scale factor. The experiment results are used to validate the effectiveness of the scaled convex hull and its simplicity.

Acknowledgments

This work was supported in part by the National Nature Science Foundation Project (61071136, 60672060, 60964001,) and Science Foundation Project of Guangxi Province (0991019Z) and Information and Communication Technology Key Laboratory Foundation Project of Guangxi Province (01902).

References

1. Brefeld, U., Geibel, P., Wysotzki, F.: Support vector machines with example dependent costs. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 23–34. Springer, Heidelberg (2003); Int. Conf. Machine Learning, pp. 57–64 (2000)
2. Domingos, P.: MetaCost: a general method for making classifiers cost-sensitive. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, pp. 155–164 (1999)

3. Elkan, C.: The foundations of cost-senstive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence, Seattle, WA, pp. 973–978 (2001)
4. Margineantu, D.D., Dietterich, T.G.: Bootstrap methods for the cost-sensitive evaluation of classifiers. In: Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, pp. 583–590 (2000)
5. Ting, K.M.: A comparative study of cost-sensitive boosting algorithms. In: Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, pp. 983–990 (2000)
6. Bradford, J.P., Kuntz, C., Kohavi, R., Brunk, C., Brodley, C.E.: Pruning decision trees with misclassification costs. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 131–136. Springer, Heidelberg (1998)
7. Knoll, U., Nakhaeizadeh, G., Tausend, B.: Cost-sensitive pruning of decision trees. In: Lavrač, N., Wrobel, S. (eds.) ECML 1995. LNCS, vol. 912, pp. 383–386. Springer, Heidelberg (1995)
8. Ting, K.M.: An instance-weighting method to induce cost-sensitive trees. IEEE Transactions on Knowledge and Data Engineering 14(3), 659–665 (2002)
9. Kukar, M., Kononenko, I.: Cost-sensitive learning with neural networks. In: Proceedings of the 13th European Conference on Artificial Intelligence, Brighton, UK, pp. 445–449 (1998)
10. Lawrence, S., Burns, I., Back, A., Tsoi, A.C., Giles, C.L.: Neural network classification and prior class probabilities. In: Orr, G.B., Müller, K.-R. (eds.) NIPS-WS 1996. LNCS, vol. 1524, pp. 299–313. Springer, Heidelberg (1998)
11. Cortes, C., Vapnik, V.N.: Support Vector Networks. Mach. Learn. 20(3), 273–297 (1995)
12. Bennett, K.P., Bredensteiner, E.J.: Duality and Geometry in SVM classifiers. In: Proc. 17th Int. Conf. Machine Learning, pp. 57–64 (2000)
13. Tao, Q., Wu, G.W., Wang, J.: A general soft method for learning SVM classifiers with L1-norm penalty. Pattern Recognit. 41(3), 939–948 (2008)
14. Keerthi, S., Shevade, S., Bhattacharyya, C., Murthy, K.: Improvements to Platt's SMO algorithm for SVM classifier design, Dept of CSA, IISc, Bangalore, India, Tech. Rep. (1999)

Erratum: Decision-Making in Drosophila with Two Conflicting Cues

Kuijie Cai[1,2], Jihong Shen[2], and Si Wu[1]

[1] Institute of Neuroscience, Shanghai Institutes for Biological Sciences,
Chinese Academy of Sciences, Shanghai 200031, China
[2] College of Science, Harbin Engineering University, Harbin 150001, China
{kjcai,siwu}@ion.ac.cn, shenjihong@hrbeu.edu.cn

D. Liu et al. (Eds.): ISNN 2011, Part I, LNCS 6675, pp. 93–100, 2011.
© Springer-Verlag Berlin Heidelberg 2011

DOI 10.1007/978-3-642-21105-8_73

The affiliations at the top of page 93 of this publication are in the wrong order. Harbin Engineering University should have been placed first, as shown below:

Kuijie Cai[1,2], Jihong Shen[1], and Si Wu[2]

[1] College of Science, Harbin Engineering University, Harbin 150001, China
[2] Institute of Neuroscience, Shanghai Institutes for Biological Sciences,
Chinese Academy of Sciences, Shanghai 200031, China
{kjcai,siwu}@ion.ac.cn, shenjihong@hrbeu.edu.cn

The original online version for this chapter can be found at
http://dx.doi.org/10.1007/978-3-642-21105-8_12

Author Index