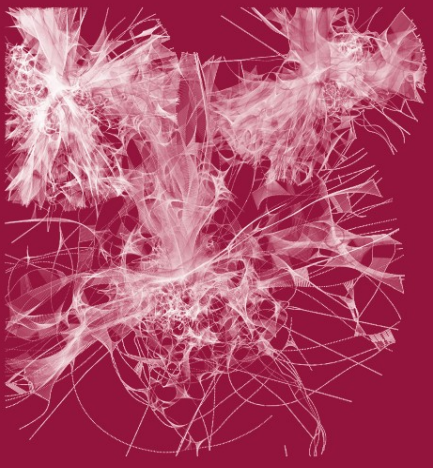
Clara Pizzuti
Marylyn D. Ritchie
Mario Giacobini (Eds.)

LNCS 5483

Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics

7th European Conference, EvoBIO 2009
Tübingen, Germany, April 2009
Proceedings



Springer

Lecture Notes in Computer Science 5483

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

Clara Pizzuti Marylyn D. Ritchie
Mario Giacobini (Eds.)

Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics

7th European Conference, EvoBIO 2009
Tübingen, Germany, April 15-17, 2009
Proceedings

Springer

Volume Editors

Clara Pizzuti
Institute for High Performance Computing and Networking (ICAR)
Italian National Research Council (CNR)
Via P. Bucci 41C, 87036 Rende (CS), Italy
E-mail: pizzuti@icar.cnr.it

Marylyn D. Ritchie
Vanderbilt University
Center for Human Genetics Research
Department of Molecular Physiology & Biophysics
519 Light Hall, Nashville, TN 37232, USA
E-mail: ritchie@chgr.mc.vanderbilt.edu

Mario Giacobini
University of Torino, Faculty of Veterinary Medicine
Department of Animal Production, Epidemiology and Ecology
GECO - Computational Epidemiology Group
Via Leonardo da Vinci 44, 10095 Grugliasco (TO), Italy
and Molecular Biotechnology Center
CBU - Computational Biology Unit
Via Nizza 52, 10126 Torino, Italy
E-mail: mario.giacobini@unito.it

Cover illustration: Detail of *You Pretty Little Flocker* by Alice Eldridge (2008)
www.infotech.monash.edu.au/research/groups/cema/flocker/flocker.html

Library of Congress Control Number: Applied for

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

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

ISSN 0302-9743
ISBN-10 3-642-01183-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-01183-2 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12652396 06/3180 5 4 3 2 1 0

Preface

The field of bioinformatics has two main objectives: the creation and maintenance of biological databases, and the discovery of knowledge from life sciences data in order to unravel the mysteries of biological function, leading to new drugs and therapies for human disease. Life sciences data come in the form of biological sequences, structures, pathways, or literature. One major aspect of discovering biological knowledge is to search, predict, or model specific information in a given dataset in order to generate new interesting knowledge. Computer science methods such as evolutionary computation, machine learning, and data mining all have a great deal to offer the field of bioinformatics. The goal of the 7th European Conference on Evolutionary Computation, Machine Learning, and Data Mining in Bioinformatics (EvoBIO 2009) was to bring together experts in these fields in order to discuss new and novel methods for tackling complex biological problems.

The 7th EvoBIO conference was held in Tübingen, Germany during April 15–17, 2009 at the Eberhard-Karls-Universität Tübingen. EvoBIO 2009 was held jointly with the 12th European Conference on Genetic Programming (EuroGP 2009), the 9th European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP 2009), and the Evo Workshops. Collectively, the conferences and workshops are organized under the name Evo* (`www.evostar.org`).

EvoBIO, held annually as a workshop since 2003, became a conference in 2007 and it is now the premiere European event for those interested in the interface between evolutionary computation, machine learning, data mining, bioinformatics, and computational biology. All papers in this book were presented at EvoBIO 2009 and responded to a call for papers that included topics of interest such as biomarker discovery, cell simulation and modeling, ecological modeling, fluxomics, gene networks, biotechnology, metabolomics, microarray analysis, phylogenetics, protein interactions, proteomics, sequence analysis and alignment, and systems biology. A total of 44 papers were submitted to the conference for peer-review. Of those, 17 (38.6%) were accepted for publication in these proceedings.

We would first and foremost like to thank all authors who spent the time and effort to produce interesting contributions to this book. We would like to thank the members of the Program Committee for their expert evaluation of the submitted papers, Jennifer Willies, for her tremendous administrative help and coordination, Ivanoe De Falco, for his fantastic work as the Publicity Chair, and Marc Ebner for his outstanding work as the local organizer. Moreover, we would like to thank the following persons and institutes: Andreas Zell, Chair of Computer Architecture at the Wilhelm Schickard Institute for Computer Science at the University of Tübingen for local support; Peter Weit, Vice Director of the

Seminar for Rhetorics at the New Philology Department at the University of Tübingen; the Tübingen Info Office, especially Marco Schubert for local tourism and information; the German Research Foundation (DFG) for financial support of the EvoStar conference; and Marc Schoenauer and the MyReview team (`http://myreview.lri.fr/`) for providing the conference review management system and efficient assistance.

Finally, we hope that you will consider contributing to EvoBIO 2010.

February 2009

Clara Pizzuti
Marylyn D. Ritchie

Organization

EvoBIO 2009, together with EuroGP 2009, EvoCOP 2009, and EvoWorkshops 2009, was part of EVO* 2009, Europe's premier co-located events in the field of evolutionary computing.

Program Chairs

Clara Pizzuti	Institute of High-Performance Computing and Networking National Research Council (ICAR-CNR), Italy
Marylyn D. Ritchie	Vanderbilt University in Nashville, USA

Local Chair

Marc Ebner	Eberhard-Karls-Universität Tübingen, Germany

Publicity Chair

Ivanoe De Falco	Institute of High-Performance Computing and Networking National Research Council (ICAR-CNR), Italy

Proceedings Chair

Mario Giacobini	University of Torino, Italy

Steering Committee

David W. Corne	Heriot-Watt University, Edinburgh, UK
Elena Marchiori	Radboud University, Nijmegen, The Netherlands
Carlos Cotta	University of Malaga, Spain
Jason H. Moore	Dartmouth Medical School in Lebanon, NH, USA
Jagath C. Rajapakse	Nanyang Technological University, Singapore

Program Committee

Jesus S. Aguilar-Ruiz	Universidad Pablo de Olavide, Spain
Wolfgang Banzhaf	Memorial University of Newfoundland, Canada
Jacek Blazewicz	Poznan University of Technology, Poland
Carlos Cotta	University of Malaga, Spain
Federico Divina	Pablo de Olavide University, Spain
Maggie Eppstein	University of Vermont, USA
Gary Fogel	Natural Selection, Inc., USA
Paolo Frasconi	Univ. degli Studi di Firenze, Italy
Alex Freitas	University of Kent,UK
Raffaele Giancarlo	University of Palermo, Italy
Rosalba Giugno	University of Catania, Italy
Lutz Hamel	University of Rhode Island, USA
Jin-Kao Hao	University of Angers, France
Tom Heskes	Radboud University Nijmegen, The Netherlands
Katharina Huber	University of East Anglia, UK
Mehmet Koyuturk	Case Western Reserve University, USA
Natalio Krasnogor	University of Nottingham, UK
Bill Langdon	Essex University, UK
Michael Lones	University of York, UK
Bob MacCallum	Imperial College London, UK
Daniel Marbach	Ecole Polytechnique Fédérale de Lausanne, Switzerland
Elena Marchiori	Radboud University, Nijmegen, The Netherlands
Andrew Martin	University College London, UK
Jason Moore	Dartmouth College, USA
Pablo Moscato	The University of Newcastle, Australia
Vincent Moulton	University of East Anglia, UK
See-Kiong Ng	Institute for Infocomm Research, Singapore
Carlotta Orsenigo	Politecnico di Milano, Italy
Clara Pizzuti	ICAR-CNR, Italy
Michael Raymer	Wright State University, USA
Marylyn Ritchie	Vanderbilt University, USA
Raul Giraldez Rojo	Pablo de Olavide University, Spain
Simona Rombo	Universitá della Calabria Italy
Jem Rowland	Aberystwyth University, UK
Marc Schoenauer	LRI- Université Paris-Sud, France
Ugur Sezerman	Sabanci University, Turkey
Marc L. Smith	Vassar College, USA
El-Ghazali Talbi	Univ. des Sciences et Technologies de Lille, France

Dave Trudgian	University of Exeter, UK
Alfonso Urso	ICAR-CNR, Italy
Antoine van Kampen	Universiteit van Amsterdam, The Netherlands
Carlo Vercellis	Politecnico di Milano, Italy
Gwenn Volkert	Kent State University, USA
Tiffani Williams	Texas A&M University, USA
Kai Ye	Leiden Amsterdam Center for Drug Research, The Netherlands
Andreas Zell	University of Tübingen, Germany
Blaz Zupan	University of Ljubljana, Slovenia

Sponsoring Institutions

- Eberhard-Karls-Universität Tübingen
- German Research Foundation (DFG)
- The Centre for Emergent Computing, Edinburgh Napier University, UK
- The Institute for High Performance Computing and Networking of the National Research Council (ICAR-CNR), Italy

Table of Contents

EvoBIO Contributions

Association Study between Gene Expression and Multiple Relevant Phenotypes with Cluster Analysis

Zhenyu Jia*, Yipeng Wang, Kai Ye, Qilan Li, Sha Tang, Shizhong Xu, and Dan Mercola

Department of Pathology and Laboratory Medicine, University of California, Irvine, CA 92697, USA
zjia@uci.edu
The Sidney Kimmel Cancer Center, San Diego, CA 92121, USA
ywang@skcc.org
European Bioinformatics Institute, Hinxton, Cambridge, CB10 1SD, United Kingdom
kye@ebi.ac.uk
Division of Medicinal Chemistry, Leiden University, 2300 RA Leiden, The Netherlands
q.li@lacdr.leidenuniv.nl
Department of Pediatrics, University of California, Irvine, CA 92697, USA
shat@uci.edu
Department of Botany and Plant Sciences, University of California, Riverside, CA 92521, USA
shxu@ucr.edu
Department of Pathology and Laboratory Medicine, University of California, Irvine, CA 92697, USA
dmercola@uci.edu

Abstract. A complex disease is usually characterized by a few relevant disease phenotypes which are dictated by complex genetical factors through different biological pathways. These pathways are very likely to overlap and interact with one another leading to a more intricate network. Identification of genes that are associated with these phenotypes will help understand the mechanism of the disease development in a comprehensive manner. However, no analytical model has been reported to deal with multiple phenotypes simultaneously in gene-phenotype association study. Typically, a phenotype is inquired at one time. The conclusion is then made simply by fusing the results from individual analysis based on single phenotype. We believe that the certain information among phenotypes may be lost by not analyzing the phenotypes jointly. In current study, we proposed to investigate the associations between expressed genes and multiple phenotypes with a single statistics model. The relationship between gene expression level and phenotypes is described by a multiple linear regression equation. Each regression coefficient, representing gene-phenotype(s) association strength, is assumed to be sampled from a mixture of two normal distributions. The two normal components are used to model the behaviors of phenotype(s)-relevant

* To whom all correspondence should be addressed.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 1–12, 2009.

genes and phenotype(s)-irrelevant genes, respectively. The conclusive classification of coefficients determines the association status between genes and phenotypes. The new method is demonstrated by simulated study as well as a real data analysis.

Keywords: Association, classification, gene expression, multiple phenotypes.

1 Introduction

Intensive efforts have been tried to identify genes that are associated with disease phenotype of interest [1,2]. Sorting out the phenotype associated genes will help understand the mechanism of the disease and develop efficient treatment to the disease. However, a complex disease is usually characterized by more than one phenotypes reflecting different facets of the disease. For example, Mini-Mental State Examination (MMSE) and Neurofibrillary Tangle count (NFT) are often used for quantifying the severity of the Alzheimer's Disease. Another example is that obesity is commonly defined as a body mass index (BMI, weight divided by height squared) of $30kg/m^2$ or higher. However, other indices, such as waist circumference, waist hip ratio and body fat percent, can also be applied as obesity indicators. Some of the disease phenotypes may be well correlated to one another but are not necessarily controlled by the same biological pathway. The gene networks involved in the development of the phenotypes may overlap and interact via the common genes shared by those networks. It is very useful to uncover genes representing the involved pathways as well as the cross-talk between those pathways.

The typical approach is to find the genes that are associated with individual phenotype first, and then combine the findings based on separate analysis. Genes shared by different phenotypes can be regarded as the 'nodes' between the participating pathways. Several methods can be used for the single-trait association analysis. The most simple way is to calculate the Pearson's correlation for each gene and the phenotype [1,3]. Genes are then sorted on the basis of the magnitude of the correlation coefficients. The genes on the top of the list are claimed to be associated with the phenotype. [4] developed a simple linear regression model to detect genes that are related to the phenotype. Subsequently, [5] and [6] enhanced the prototype to take into account the non-linear association between the gene expression levels and the phenotypic values. In those model-based methods, the gene expression level is described as a linear function of phenotypic value. The significance level of the regression coefficient is suggestive of the strength of the association between gene and phenotype. Classification approach is then utilized to cluster genes based on the magnitude of the regression coefficients. With the linear-model-based method, the association study has been turned into a classification problem. To our best knowledge, no analytical model has been reported to deal with multiple phenotypes simultaneously in such association study. We consider that, to some extent, information among phenotypes may be

lost in single-trait method since phenotypes are analyzed separately, and such missing information is important for inferring the interaction between relevant phenotypes.

In current study, we developed an advanced model which analyzes the association between genes and multiple phenotypes jointly. The expression level of a gene is described as a linear regression function of the phenotypes and their interactions. The gene specific regression coefficient, either main effect or interaction, is assumed to be sampled from a mixture of two normal distributions. One normal component has mean zero and a very tiny variance; while the other normal component also has mean zero but a much larger variance. If the coefficient is from the first normal component, then its size is close to zero and there is no indication of association; otherwise, if the coefficient is from the second normal component, the size of the coefficient is nontrivial suggesting the association between the gene and the phenotype(s). Such mixture model setting originated from the spike and slab distribution proposed by [7]. We compared the new method with other single-trait approaches by synthesized data as well as real data generated from microarray experiment. The new method appeared to be more desirable for gene-phenotype association study.

2 Methods

2.1 Linear Regression Setting and Bayesian Modelling

Let Z_{js} be the standardized phenotypic value for trait j and subject s, where $j = 1, \ldots, p$ and $s = 1, \ldots, n$. Here all the phenotypes are normalized under the same scale, *i.e.*, $[-1, 1]$. For example, suppose $\Psi_j = [\Psi_{j1}, \ldots, \Psi_{jn}]$ is the original measurements for trait j, and if Ψ_{js} is a continuous variable, then

$$Z_j = [Z_{j1}, \ldots, Z_{jn}] = 2 \times \frac{\Psi_j - \mathrm{Min}(\Psi_j)}{\mathrm{Max}(\Psi_j) - \mathrm{Min}(\Psi_j)} - 1.$$

If Ψ_{js} is a categorical phenotype, we use $Z_{js} = \{-1, 1\}$ for binary variable or $Z_{js} = \{-1, 0, 1\}$ for trinary variable. In current study, only binary or trinary variable are considered for simplicity. Let $Y_i = [Y_{i1}, \ldots, Y_{in}]$ be the expression levels of gene i, where $i = 1, \ldots, m$. We may use following linear model to describe the relationship between the gene expression and phenotypes,

$$Y_i = b_{i0} + \sum_{j=1}^{p} Z_j b_{ij} + e_i, \tag{1}$$

where b_{i0} is the intercept, b_{ij} is the regression coefficient for Z_j, and $e_i = [e_{i1}, \ldots, e_{in}]$ is random error with multivariate normal distribution $\mathrm{N}(\mathbf{0}, I\sigma^2)$. Note that the number of genes is usually much greater than that of phenotypes. So, it is plausible to regress gene expression on phenotypes to reduce the

dimension of parameters. If the interactions between the phenotypes are also considered, model 1 can be rewritten as

$$Y_i = b_{i0} + \sum_{j=1}^{p} Z_j b_{ij} + \sum_{j=1}^{p-1} \sum_{k>j}^{p} Z_j Z_k b_{ijk} + e_i. \tag{2}$$

For simplicity, we only consider the 2-way interaction in current study. If we treat the interaction terms as ordinary regression terms, there will be a total of $(p^2 + p)/2$ regression terms in the linear model. We can further rewrite model 2 as

$$Y_i = \beta_{i0} + \sum_{l=1}^{(p^2+p)/2} X_l \beta_{il} + e_i, \tag{3}$$

where β's represent b series, and X's represent Z series. Note that β_0 is irrelevant to the association study. It can be simply removed from model via a linear contrasting scheme, *i.e.*,

$$y_i = \sum_{l=1}^{(p^2+p)/2} x_l \beta_{il} + \varepsilon_i, \tag{4}$$

where $\sum_{s=1}^{n} y_{is} = 0$ and $\sum_{s=1}^{n} x_{js} = 0$. After linear contrasting transformation, the n pieces of information for each gene are no longer independent to one another because the constraint of "sum-to-zero" has been imposed to y_i and x_l. Therefore, the last element of vector y_i should be removed and y_i becomes an $N \times 1$ vector for $N = n - 1$. Accordingly, x_l becomes an $N \times 1$ vector, and ε_i is now $\mathrm{N}(\mathbf{0}, R\sigma^2)$ distributed with a known $N \times N$ positive definite matrix R (see [5] for more details).

To simplify the presentation, we use the following notation to express different probability densities throughout the current study:

$$p(\text{variable}|\text{parameter list}) = \text{DensityName}(\text{variable}; \text{parameter list}).$$

We assume that each regression coefficient β_{ij} in model 4 is sampled from the following two-normal-components-mixture distribution:

$$p(\beta_{il}|\eta_{il}, \sigma_l^2) = (1 - \eta_{il})\text{Normal}(\beta_{il}; 0, \delta) + \eta_{il}\text{Normal}(\beta_{il}; 0, \sigma_l^2), \tag{5}$$

where the two normal distributions are both centered at zero but have different levels of spread. Such mixture setting was originally suggested by [8]. We use the first normal distribution to model trivial effects by enforcing a fixed tiny variance $\delta = 10^{-2}$; while, for the second normal component, a larger variance σ_l^2 is utilized to govern nonzero effects. The unobserved variable $\eta_{il} = \{0, 1\}$ is used for indicating whether β_{il} is sampled from $\mathrm{N}(0, \delta)$ or $\mathrm{N}(0, \sigma_l^2)$. Our goal is to infer the posterior distribution of η_{il} and estimate the likelihood of association. We further describe η_{il} by a Bernoulii distribution with probability ρ_l, denoted by

$$p(\eta_{il}|\rho_l) = \text{Bernoulii}(\eta_{il}; \rho_l). \tag{6}$$

The parameter ρ_l and its counterpart $1-\rho_l$, which can be viewed as mixing proportion of the two-component mixture model, regulate the number of genes that are connected with the lth regression term. This parameter ρ_l is *per se* governed by a Dirichlet distribution, denoted by $\text{Dirichlet}(\rho_l; 1, 1)$. The variance components of the hierarchical model are assigned with scaled inverse chi-square distributions, denoted by $\text{Inv}-\chi^2(\sigma_l^2; d_0, \omega_0)$. We chose $d_0 = 5$ and $\omega_0 = 50$ for σ_l^2, and chose $d_0 = 0$ and $\omega_0 = 0$ for σ^2. The hyper-parameters were selected based on our previous experience of similar analyses.

2.2 Markov Chain Monte Carlo

The inference of the parameters of interest is accomplished by using MCMC sampling, which is a algorithm for sampling from probability distributions based on constructing a Markov chain that has the desired distribution as its equilibrium distribution. We initialize all the parameters at very first step. Then the probability distribution of any parameter given the other parameters is derived. The value of the parameter is then replaced by a sample drawn from the obtained distribution. Such process is repeated sequentially until all the parameters have been updated. We call it a sweep if we finish updating all the parameters for one time. A Markov chain is constructed by sweeping for many thousands of times. The state of the chain after a large number of sweeps is then used as a sample from the desired distribution. The detailed sampling scheme for each variable is described as follows.

(1) Variable η_{il} is drawn from $\text{Bernoulli}(\eta_{il}; \pi_{il})$, where

$$\pi_{il} = \frac{\rho_l \text{N}(\beta_{il}; 0, \sigma_l^2)}{\rho_l \text{N}(\beta_{il}; 0, \sigma_l^2) + (1-\rho_l)\text{N}(\beta_{il}; 0, \delta)} \tag{7}$$

(2) Variable β_{il} is drawn from $\text{N}(\beta_{il}; \mu_\beta, \sigma_\beta^2)$, where

$$\mu_\beta = \left[x_l^T R^{-1} x_l + \frac{\sigma^2}{\eta_{il}\sigma_l^2 + (1-\eta_{il})\delta} \right]^{-1} x_l^T R^{-1} \triangle y_i, \tag{8}$$

$$\sigma_\beta^2 = \left[x_l^T R^{-1} x_l + \frac{\sigma^2}{\eta_{il}\sigma_l^2 + (1-\eta_{il})\delta} \right]^{-1} \sigma^2 \tag{9}$$

and

$$\triangle y_i = y_i - \sum_{l' \neq l}^{(p^2+p)/2} x_{l'} \beta_{il'}. \tag{10}$$

(3) Sample σ_l^2 from

$$\text{Inv}-\chi^2 \left(\sigma_l^2; \sum_{i=1}^{m} \eta_{il} + 5, \sum_{i=1}^{m} \eta_{il}\beta_{il}^2 + 50 \right).$$

(4) Sample σ^2 from

$$\text{Inv}-\chi^2\left(\sigma^2; mn, \sum_{i=1}^{m}(y_i - \sum_{l=1}^{(p^2+p)/2} x_l\beta_{il})^T R^{-1}(y_i - \sum_{l=1}^{(p^2+p)/2} x_l\beta_{il})\right).$$

(5) Sample ρ_l from

$$\text{Dirichlet}\left(\rho_l; \sum_{i=1}^{m}\eta_{il}+1, m-\sum_{i=1}^{m}\eta_{il}+1\right).$$

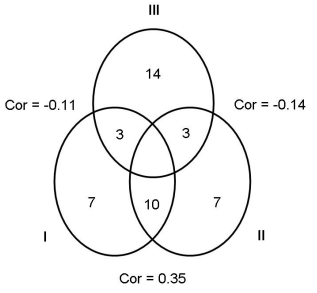
Usually it is not hard to construct a Markov Chain with the desired properties. The more difficult problem is to determine how many sweeps are needed to converge to the stationary distribution within an acceptable error. Diagnostic tool, such as the R package "coda" [9], can be used to estimate the required length of the chain for convergency. Once the chain has converged, the burn-in iterations from the beginning of the chain are discard; while for the remaining portion of the chain, one sample in every 10 sweeps is saved to form a posterior sample for statistical inference.

3 Experimental Analysis

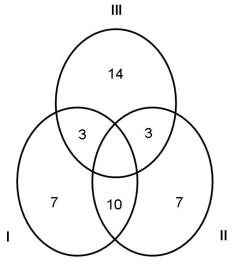
3.1 Simulated Study

In simulated study, we generated three phenotypes (I, II and III) for 100 individuals. Of the three phenotypes, the first two are correlated while the third is relatively irrelevant to the other two. The correlations between these pseudo-phenotypes are given in Figure 1(a). Such setting was meant to mimic the situation we met with in real data analysis (see next subsection). We also generated 1000 genes for each individual based on the phenotypic values we simulated. Of the 1000 genes, 20 are associated with phenotype I, 20 are associated with phenotype II, and 20 are associated with phenotype III. Note that the genes associated with different phenotypes are not mutually exclusive. The relationship between the genes associated with the three phenotypes are shown in Figure 1(a). For simplicity, all main effects and interaction effects are set to 1, and the residual variance is set to 0.01.

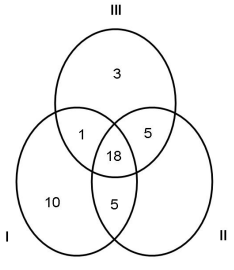
We first used the new method (denoted by method 1) to analyze the simulated data. All of the 44 true associated genes were identified and placed in the right positions of Venn Diagram (Figure 1(b)). We then used single-trait methods of [4] (denoted by method 2) and [6] (denoted by method 3) to analyze the same data set. The results are shown in Figure 1(c) and Figure 1(d), respectively. Only 41 true associated genes were identified by single-trait methods 2 and 3. Besides, many spurious associations have been claimed by methods 2 and 3. For example, no simulated genes are associated with all three phenotypes; however, methods 2 and 3 declared more than 10 genes shared by the three phenotypes. We define a linkage as a true gene-phenotype association, and a



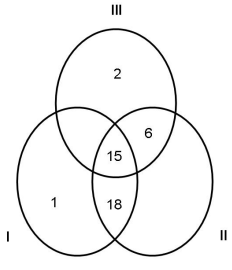
(a)



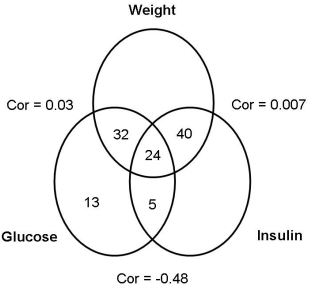
(b)



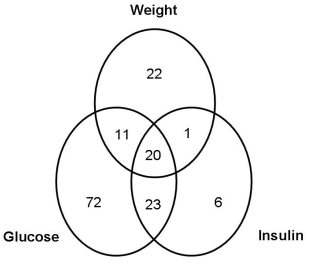
(c)



(d)



(e)



(f)

Fig. 1. Venn diagrams. (a) The numbers of genes related to the 3 simulated phenotypes and the pairwise correlations of the 3 simulated phenotypes. (b) The numbers of genes declared to be relevant to the 3 simulated phenotypes by method 1. (c) The numbers of genes declared to be relevant to the 3 simulated phenotypes by method 2. (d) The numbers of genes declared to be relevant to the 3 simulated phenotypes by method 3. (e) The numbers of genes declared to be relevant to the 3 obesity phenotypes by method 1 and the pairwise correlations of the 3 obesity phenotypes. (f) The numbers of genes declared to be relevant to the 3 obesity phenotypes by method 3.

Table 1. Type I and type II error rates for three analytical methods in simulated studies

	Method		
	1	2	3
α	0.000	0.300	0.100
β	0.000	0.016	0.014

non-linkage if gene is not related to phenotype. Therefore, a total of 60 linkages and 2940 ($1000 \times 3 - 60$) non-linkages have been generated for the simulated study. Numerically, the empirical Type I error is defined as

$$\alpha = \frac{\text{Number of undetected linkages}}{\text{Total number of linkages}},$$

and empirical Type II error is defined as

$$\beta = \frac{\text{Number of declared non-linkages}}{\text{Total number of non-linkages}}.$$

The type I and type II error rates for three methods are listed in Table 1. The new method, which considers multiple traits jointly, appeared superior to the single-trait methods 2 and 3. For single-trait analysis, we have one more time proved that the model based on non-linear association outperforms the model of simple linear association. (The comparison between methods 2 and 3 was originally presented in [6]). We additionally noted that, for single-trait methods 2 and 3, many genes are in common for phenotypes I and II because these two phenotypes are highly correlated (cor = 0.35 and p value = 0.00038). Genes that are authentically linked to a phenotype are likely to be claimed being associated with another correlated phenotype in single-trait analysis. This reasoning can also be supported by the phenomena that the number of genes shared by phenotypes II and III tended to be larger than the number shared by phenotypes I and III and the correlation between phenotypes II and III (cor = -0.14) is slightly larger than that between phenotypes I and III (cor = -0.11).

We simulated 19 more data sets with the same setting and repeated the analysis by 19 times. The results are almost identical to the presented one (data not shown). We then varied the residual variance from 0.01 to 0.81, and used the new method to analyze the additional simulated data sets. The analytical results are given in Table 2. It seems that the type II error (β) inflates faster than type I error (α) as residual variance increases.

3.2 Analysis of Mice Data

To demonstrate the new method, we used a mice data generated for obesity study [2]. In this study, Affymetrix GeneChip Mouse Expression Array was used to survey the expression levels of more than 40,000 transcript for 60 ob/ob mice.

Table 2. Type I and type II error rates for the new method when different residual variances were used for simulated studies

	Residual variance σ^2								
	0.01	0.04	0.09	0.16	0.25	0.36	0.49	0.64	0.81
α	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
β	0.00	0.00	0.00	0.00	0.00	0.11	0.26	0.43	0.58

Table 3. The numbers of genes identified by method 1, method 3 and both

	Method		
	1	3	1&3
Glu	74	126	60
Ins	69	50	19
Wt	96	54	33
All	114	155	79

A total of 25 obesity related phenotypes were collected for the experimental units. The data is publicly available at gene expression omnibus (GEO) with accession no. GSE3330. We first removed invariant genes to lessen the computational burden. About 5000 transcripts with variance greater than 0.05 were saved for further analysis. Similar pre-screening scheme has been used for array data analyses [10,11]. For the sake of convenience, we only considered three phenotypes that have been often examined in related researches. These three phenotypes were plasma glucose level (mg/dl), plasma insulin level (ng/ml) and body weight. The phenotypic data were collected at eight weeks of age. In this subsection, we only compared the new multiple-traits method (method 1) to single-trait method 3, since method 3 has been repeatedly verified being more efficient than other single-trait approaches, such as method 2. We used methods 1 and 3 to analyze this mice data and the results are shown in Figure 1(e) and Figure 1(f), respectively. Note that glucose level (Glu) is correlated with insulin level (Ins); while there is no obvious correlation between body weight (Wt) and these two traits. The three phenotypes for simulated study (see previous subsection) were generated in concordance to this observation. Comparing Figure 1(e) with Figure 1(f), we found that a lot more genes were declared to be associated with both Glu and Ins by method 3, which is no wonder since these two phenotypes are well correlated. Similar false linkages due to correlation between phenotypes have already been noticed in simulated study. We compared the gene identity of genes identified by both methods (Table 3). The results from two analyses are essentially consistent.

4 Discussion

Gene-phenotype association analysis is a pilot study for disclosing the biological process of disease development. Typically, data on more than one trait are

collected in disease-related studies. More often than not, these disease phenotypes are correlated indicating commonality of the pathways responsible for those phenotypes. However, the existing analytical tools solely deal with a phenotype at one time; thus, potential information shared by multiple phenotypes are bound to be lost. We consider that such missing information is important for inferring the interaction between correlated phenotypes and it can be picked up only through a joint analysis of multiple traits. For the first time, we proposed to analyze the associations between gene expressions and multiple phenotypes in a single statistical model. In simulated study, the new multiple-traits method appeared superior to the single-trait methods.

In real data analysis, more genes have been detected by method 3 [6] than the new method. This is because method 3 considers the gene-phenotype association beyond linearity; while, for simplicity in current study, only main effects (first order or linear association) and 2-way interaction are taken into account for the new method. However, the new method can easily extend to include higher order of main effects and interactions. The enhanced model should be more powerful for discovering relevant genes. We contrasted our analytical findings (by methods 1 and 3) with the obesity related genes obtained from NCBI. Only a small portion of identified genes have been previously reported, leaving a large number of newly detected genes worthy of a closer look in the future researches.

Mapping quantitative trait loci (QTL) has been widely used for inferring genomic loci that dictate phenotypes of interest [12,13,14,15,16,17,18,19]. However, QTL analysis are different from gene-phenotype association analysis discussed in the current study. Data for QTL analysis consist of phenotypic data and genotypic data (markers and their genotypes); while gene-phenotype association analysis involve gene expression data and phenotypic data. For quite a long time, QTL mapping also has been limited to single-trait scheme. Recently, the first multiple-traits QTL method was developed by using multivariate linear regression models [20]. Nevertheless, this multiple-traits QTL method can not be directly adopted by gene-phenotype association analysis. Usually, the dimension of gene expression data is dramatically larger than that of genotypic data. It seems impractical to regress phenotypes (at most of tens) on expression of many thousands of genes. Such model, even if it worked for a simple regression setting, would be very difficult to include higher order effects. It is natural to reverse the roles of phenotypes and genes by regressing gene expression on phenotypes. With the workable dimension of predictive variables, interactions between phenotypes or underlying pathways can be easily appreciated by adding in cross-product terms. On the other hand, clustering along gene coordinate allows effective information sharing between correlated genes.

If only gene expression data and genotypic data have been collected from biological experiment, one may carry out expression quantitative trait loci (eQTL) mapping to find likely loci that account for the variations of expression traits or gene expression [21,22,23]. Single-trait QTL approaches are typically utilized for eQTL analysis since the numbers of expression traits is far beyond the capacity of current multiple-traits analysis. Obviously, potential gene-gene correlations have

been overlooked by analyzing each gene separately. To solve this problem, [24] and [10] developed mixture models to restore the lost information by clustering along gene coordinate. If phenotypic data, gene expression data and genotypic data are all available, we may conduct genetical mapping for expression traits and regular phenotypes jointly. Association between phenotype and expressed gene is determined if they are mapped to the same locus on the genome. In addition, gene-gene relationship, either *cis-* or *trans-*, will be clear at the same time.

Acknowledgment

This research was supported by the National Institute of Health SPECS Consortium grant CA 114810-02, and the Faculty Career Development Awards of UCI to ZJ.

References

1. Blalock, E.M., Geddes, J.W., Chen, K.C., Porter, N.M., Markesbery, W.R., Landfield, P.W.: Incipient alzheimer's disease: Microarray correlation analyses reveal major transcriptional and tumor suppressor responses. Proceedings of the National Academy of Sciences of the United States of America 101, 2173–2178 (2004)
2. Lan, H., Chen, M., Flowers, J.B., Yandell, B.S., Stapleton, D.S., Mata, C.M., Mui, E.T., Flowers, M.T., Schueler, K.L., Manly, K.F., Williams, R.W., Kendziorski, C., Attie, A.D.: Combined expression trait correlations and expression quantitative trait locus mapping. Plos Genetics 2, e6 (2006)
3. van Bakel, H., Strengman, E., Wijmenga, C., Holstege, F.C.P.: Gene expression profiling and phenotype analyses of s. cerevisiae in response to changing copper reveals six genes with new roles in copper and iron metabolism. Physiol. Genomics 22, 356–367 (2005)
4. Jia, Z., Xu, S.: Clustering expressed genes on the basis of their association with a quantitative phenotype. Genetical Research 86, 193–207 (2005)
5. Qu, Y., Xu, S.H.: Quantitative trait associated microarray gene expression data analysis. Molecular Biology and Evolution 23, 1558–1573 (2006)
6. Jia, Z., Tang, S., Mercola, D., Xu, S.: Detection of quantitative trait associated genes using cluster analysis. In: Marchiori, E., Moore, J.H. (eds.) EvoBIO 2008. LNCS, vol. 4973, pp. 83–94. Springer, Heidelberg (2008)
7. Mitchell, T.J., Beauchamp, J.J.: Bayesian variable selection in linear regression. Journal of the American Statistical Association 83, 1023–1036 (1988)
8. George, E.I., Mcculloch, R.E.: Variable selection via gibbs sampling. Journal of the American Statistical Association 88, 881–889 (1993)
9. Raftery, A.E., Lewis, S.M.: One long run with diagnostics: Implementation strategies for markov chain monte carlo. Statistical Science 7, 493–497 (1992)
10. Jia, Z., Xu, S.: Mapping quantitative trait loci for expression abundance. Genetics 176, 611–623 (2007)
11. Ghazalpour, A., Doss, S., Zhang, B., Wang, S., Plaisier, C., Castellanos, R., Brozell, A., Schadt, E.E., Drake, T.A., Lusis, A.J., Horvath, S.: Integrating genetic and network analysis to characterize genes related to mouse weight. Plos Genetics 2, 1182–1192 (2006)

12. Lander, E.S., Botstein, D.: Mapping mendelian factors underlying quantitative traits using rflp linkage maps. Genetics 121, 185–199 (1989)
13. Jiang, C.J., Zeng, Z.B.: Mapping quantitative trait loci with dominant and missing markers in various crosses from two inbred lines. Genetica 101, 47–58 (1997)
14. Xu, S.Z., Yi, N.J.: Mixed model analysis of quantitative trait loci. Proceedings of the National Academy of Sciences of the United States of America 97, 14542–14547 (2000)
15. Broman, K.W., Speed, T.R.: A model selection approach for the identification of quantitative trait loci in experimental crosses. Journal of the Royal Statistical Society Series B-Statistical Methodology 64, 641–656 (2002)
16. Yi, N.J., Xu, S.Z.: Mapping quantitative trait loci with epistatic effects. Genetical Research 79, 185–198 (2002)
17. Yi, N.J., Xu, S.Z., Allison, D.B.: Bayesian model choice and search strategies for mapping interacting quantitative trait loci. Genetics 165, 867–883 (2003)
18. Rzhetsky, A., Wajngurt, D., Park, N., Zheng, T.: Probing genetic overlap among complex human phenotypes. Proceedings of the National Academy of Sciences of the United States of America 104, 11694–11699 (2007)
19. Oti, M., Huynen, M.A., Brunner, H.G.: Phenome connections. Trends in Genetics 24, 103–106 (2008)
20. Banerjee, S., Yandell, B.S., Yi, N.: Bayesian quantitative trait loci mapping for multiple traits. Genetics 179, 2275–2289 (2008)
21. Schadt, E.E., Monks, S.A., Drake, T.A., Lusis, A.J., Che, N., Colinayo, V., Ruff, T.G., Milligan, S.B., Lamb, J.R., Cavet, G., Linsley, P.S., Mao, M., Stoughton, R.B., Friend, S.H.: Genetics of gene expression surveyed in maize, mouse and man. Nature 422, 297–302 (2003)
22. Hubner, N., Wallace, C.A., Zimdahl, H., Petretto, E., Schulz, H., Maciver, F., Mueller, M., Hummel, O., Monti, J., Zidek, V., Musilova, A., Kren, V., Causton, H., Game, L., Born, G., Schmidt, S., Muller, A., Cook, S.A., Kurtz, T.W., Whittaker, J., Pravenec, M., Aitman, T.J.: Integrated transcriptional profiling and linkage analysis for identification of genes underlying disease. Nature Genetics 37, 243–253 (2005)
23. Jansen, R.C., Nap, J.P.: Genetical genomics: the added value from segregation. Trends in Genetics 17, 388–391 (2001)
24. Kendziorski, C.M., Chen, M., Yuan, M., Lan, H., Attie, A.D.: Statistical methods for expression quantitative trait loci (eqtl) mapping. Biometrics 62(1), 19–27 (2006)

Gaussian Graphical Models to Infer Putative Genes Involved in Nitrogen Catabolite Repression in *S. cerevisiae*[⋆]

Kevin Kontos[1,⋆⋆], Bruno André[2], Jacques van Helden[3], and Gianluca Bontempi[1]

[1] Machine Learning Group, Faculté des Sciences, Université Libre de Bruxelles (ULB), Boulevard du Triomphe CP 212, 1050 Brussels, Belgium
{kkontos,gbonte}@ulb.ac.be
http://www.ulb.ac.be/di/mlg/

[2] Physiologie Moléculaire de la Cellule, IBMM, Faculté des Sciences, ULB, Rue des Pr. Jeener et Brachet 12, 6041 Gosselies, Belgium
bran@ulb.ac.be

[3] Laboratoire de Bioinformatique des Génomes et des Réseaux, Faculté des Sciences, ULB, Boulevard du Triomphe CP 263, 1050 Brussels, Belgium
Jacques.van.Helden@ulb.ac.be

Abstract. Nitrogen is an essential nutrient for all life forms. Like most unicellular organisms, the yeast *Saccharomyces cerevisiae* transports and catabolizes good nitrogen sources in preference to poor ones. Nitrogen catabolite repression (NCR) refers to this selection mechanism. We propose an approach based on Gaussian graphical models (GGMs), which enable to distinguish direct from indirect interactions between genes, to identify putative NCR genes from putative NCR regulatory motifs and over-represented motifs in the upstream noncoding sequences of annotated NCR genes. Because of the high-dimensionality of the data, we use a shrinkage estimator of the covariance matrix to infer the GGMs. We show that our approach makes significant and biologically valid predictions. We also show that GGMs are more effective than models that rely on measures of direct interactions between genes.

1 Introduction

Nitrogen is an essential nutrient for all life forms. The emergence of cells able to transport, catabolize and synthesize a wide variety of nitrogenous compounds has thus been favored by evolutionary selective pressure [1]. As a consequence, the yeast *Saccharomyces cerevisiae* can use almost 30 distinct nitrogen-containing compounds [1]. Like most unicellular organisms, yeast transports and catabolizes good nitrogen sources in preference to poor ones. Nitrogen catabolite repression

⋆ This work was supported by the Communauté Française de Belgique (ARC grant no. 04/09-307).

⋆⋆ Corresponding author.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 13–24, 2009.

(NCR) refers to this selection mechanism [1,2]. More specifically, NCR inhibits the transcriptional activation systems of genes needed to degrade poor nitrogen sources [2]. In this context, bioinformatics approaches offer the possibility to identify a relatively small number of putative NCR genes [1,3]. Hence, biologists need only to test a small number of "promising" candidates, instead of testing all genes, saving time and resources.

In this paper, we use Gaussian graphical models (GGMs) [4] to infer putative NCR genes from putative NCR regulatory motifs and over-represented motifs in the upstream noncoding sequences of annotated NCR genes. The use of counts of pattern occurrences in noncoding sequences has already been successfully used for predicting gene co-regulation [5] and for the identification of putative NCR genes [1,3]. GGMs have become increasingly popular to infer multivariate dependencies among genes in functional genomics (see, e.g., [6]). These models encode full conditional relationships between genes. Hence, they enable to distinguish direct from indirect interactions.

Standard multivariate methods for structure learning of Gaussian graphical models require the estimation of the full joint probability distribution. Unfortunately, the data sets typically encountered in functional genomics describe a large number of variables (on the order of hundreds or thousands) but only contain comparatively few samples (on the order of tens or hundreds), which renders this estimation an ill-posed problem. Hence, standard multivariate methods cannot be applied directly. Two alternative approaches have thus been introduced: one uses regularization techniques [7,6], while the other uses limited-order partial correlations [8,9,10,11]. The former approach is discussed in this paper. More specifically, we use a shrinkage covariance estimator that exploits the Ledoit-Wolf lemma [12]. This estimator, which has been successfully used to infer genetic regulatory networks (GRN) from microarray data [6], is statistically efficient and fast to compute [12].

We show that our approach makes significant and biologically valid predictions by comparing these predictions to annotated and putative NCR genes, and by performing negative controls. In particular, we found that the annotated NCR genes form a coherent set of genes in terms of partial correlations. These results suggest that our approach can successfully identify potential NCR genes in *S. cerevisiae*. We also show the GGM is more effective than the independence graph that relies on a measure of direct interaction between genes.

The outline of the paper is as follows. Section 2 briefly describes the sets of genes identified as relevant/irrelevant to NCR in *S. cerevisiae*. Section 3 introduces independence graph, Gaussian graphical models and the shrinkage covariance estimator. Our approach to inferring putative NCR genes is introduced in Sect. 4. The experimental setup is presented in Sect. 5. The results, followed by a short discussion, are given in Sect. 6. Finally, Sect. 7 concludes the paper.

2 Nitrogen Catabolite Repression in *S. cerevisiae*

Nitrogen catabolite repression (NCR) in *S. cerevisiae* consists in the specific inhibition of transcriptional activation of genes encoding the permeases and

catabolic enzymes needed to degrade poor nitrogen sources [2]. All known nitrogen catabolite pathways are regulated by four regulators (Gln3, Gat1, Dal80, and Deh1) [13]. We will denote by RNCR the set of genes coding for these regulators.

We also have available a set of 37 annotated NCR genes (ANCR) [1,3], and three sets of putative NCR genes, denoted by $\mathcal{G}$, $\mathcal{S}$ and $\mathcal{B}$, identified in three genome-wide experimental[1] studies [1,2,14], respectively. Note that the sets RNCR and ANCR partially overlap as three of the four RNCR genes are also targets of NCR. Finally, we also have a set of 89 manually-selected genes known to be insensitive to NCR (NNCR) [3].

3 Independence Graph and Gaussian Graphical Models

3.1 Overview

In an independence graph [15] (also known as relevance network), pairs of genes are connected (i.e., assumed to interact) if their respective correlation exceeds a given threshold. Independence graphs therefore represent the marginal independence structure of the genes. Although marginal independence is a strong indicator for independence, it is a weak criterion for measuring dependence, since more or less all genes will be marginally (i.e., directly or indirectly) correlated [6].

Gaussian graphical models (GGMs) [4,16,17] have become popular to infer multivariate dependencies in the "small n, large p" data setting (i.e., when the number of samples n is small compared to the number of genes p) commonly encountered in functional genomics, especially for inferring genetic regulatory networks from microarray data [6,8,18]. They have been introduced to overcome the shortcoming of the independence graph. In these models, missing edges denote zero full-order partial correlations, and therefore, correspond to conditional independence relationships.

Therefore, GGMs have an important advantage compared to independence graphs: full-order partial correlation measures the association between two genes while taking into account all the remaining observed genes. Hence, GGMs enable to distinguish direct from indirect interactions between genes due to intermediate genes or directly due to other genes.

3.2 Inferring Gaussian Graphical Models

However, computing full-order partial correlations requires the full joint distribution of genes. This is problematic in the "small n, large p" data setting: the maximum likelihood estimate of the population concentration matrix needed to infer GGMs requires that the sample covariance matrix has full rank and this holds, with probability one, if and only if $n > p$ [19].

To circumvent this problem, two alternative approaches have been introduced: one uses regularization techniques [7,6], while the other uses limited-order partial correlations [8,9,10,11]. The former approach is used in this paper. More

[1] The study by Godard et al. [1] also contains a bioinformatics validation part.

specifically, we use a shrinkage covariance estimator that exploits the Ledoit-Wolf lemma [12]. This estimator, which has been successfully used to infer GRN from microarray data [6], is statistically efficient and fast to compute [12].

3.3 Unbiased Estimator

Let X denote a $p \times n$ matrix of n independent and identically distributed observations of p random variables with mean zero and covariance matrix Σ. Let $\hat{S}$ denote the unbiased estimator of the covariance matrix Σ, i.e., the unbiased sample covariance matrix: $\hat{S} = \frac{1}{n-1} X X^t$, where X^t is the transpose of X.

This estimator exhibits high variance [12]. To decrease it, and thus also its mean squared error (MSE), we use a shrinkage estimator [12] which seeks the optimal trade-off between error due to bias and error due to variance by "shrinking" the sample covariance matrix towards a low-dimensional estimator.

3.4 Shrinkage Estimator

Let $\hat{T}$ denote the low-dimensional (biased) estimator of the covariance matrix Σ whose (i,j)-th element is defined by $\hat{t}_{ij} = \hat{s}_{ii}$ if $i = j$, and $\hat{t}_{ij} = 0$ otherwise.

The number of parameters to be fitted in the constrained estimate $\hat{T}$ is small compared to that of the unconstrained estimate $\hat{S}$ (p parameters instead of $p\,(p+1)\,/2$). Hence, the constrained estimate $\hat{T}$ will exhibit a lower variance than its unconstrained counterpart $\hat{S}$. On the other hand, the former will exhibit considerable bias as an estimator of Σ (recall that the latter is unbiased).

The linear shrinkage estimator $\hat{\Sigma}$ combines both estimators in a weighted average, instead of choosing between one of these two extremes: $\hat{\Sigma} = \lambda \hat{T} + (1 - \lambda)\hat{S}$, where $\lambda \in [0,1]$ represents the shrinkage intensity. The *optimal* shrinkage estimator $\hat{\Sigma}^*$, which we will simply refer to as the shrinkage estimator, minimizes the expected quadratic loss $\mathrm{E}\left[\|\Sigma^* - \Sigma\|_F^2\right]$, where $\|\cdot\|_F$ denotes the Frobenius norm, i.e., $\|M\|_F = \sqrt{\mathrm{tr}\,(MM^t)}$, where $\mathrm{tr}\,(\cdot)$ denotes matrix trace.

Ledoit and Wolf [12] showed that $\hat{\Sigma}^*$ is both well-conditioned and more accurate than the sample covariance matrix $\hat{S}$ asymptotically. They also showed experimentally that the asymptotic results tend to hold well in finite sample. Most importantly, they derived an analytical solution (see also [6]) to the problem of choosing the optimal value (under square loss) for the shrinkage parameter λ, therefore avoiding computationally intensive procedures such as cross-validation, bootstrap or MCMC.

The shrinkage estimator of the partial correlation matrix $\hat{\Omega}^*$ is simply the inverse of $\hat{\Sigma}^*$ [4]: $\hat{\Omega}^* = \left(\hat{\Sigma}^*\right)^{-1}$.

4 Inferring Putative NCR Genes with GGMs

4.1 Introduction

Suppose we have a set $\mathcal{C}$ of genes known (or hypothesised) to be involved in NCR. We will refer to it as the "core" set. It is either composed of the known

NCR regulators, $\mathcal{C} = \text{RNCR}$, or the set of annotated NCR genes, $\mathcal{C} = \text{ANCR}$ (see Sect. 5).

Further suppose we have a $p \times n$ data matrix X, where p is the number of genes (variables) and n is the number of samples (see Sect. 3.3). Hence, each sample is a p-dimensional vector. The samples reflect properties which are known (or hypothesised) to be relevant for NCR. In this paper, samples correspond to motifs relevant for the NCR regulation [1] (see Sect. 4.2). Hence, the (i, j)-th entry of matrix X, denoted x_{ij}, represents the number of occurrences of motif j in the upstream noncoding sequence of gene i.

4.2 Data Set

Based on prior biological knowledge, we defined a set of 9 motifs as potentially relevant for the NCR regulation (see [1]). We also used the program *oligo-analysis* from RSAT [20] to detect over-represented oligonucleotides (for all sizes between 5 and 8) in the promoter sequences of the 37 ANCR genes, leading to a total of 56 significantly over-represented oligonucleotides [1]. Since some annotated motifs were also detected by *oligo-analysis*, we generated a non-redundant list of $n = 62$ motifs of interest. Finally, the program *dna-pattern* [20] was used to count the occurrences of the $n = 62$ motifs in each of the $p = 5869$ yeast gene promoters (note that $p \gg n$).[2]

4.3 Method

The proposed approach consists in inferring (linear) dependencies between yeast genes by computing (full-order) partial correlations from the data matrix X using the shrinkage estimator presented in Sect. 3.4. These multivariate dependencies are then exploited by selecting the genes that are correlated (in terms of partial correlation) with at least one gene of the "core" set $\mathcal{C}$.

More specifically, for a given threshold t, the set $\mathcal{I}_t$ of inferred NCR genes is given by:

$$\mathcal{I}_t = \left\{ j \in \mathcal{A} \setminus \mathcal{C} : \max_{i \in \mathcal{C}} \left| \hat{\omega}^*_{ij} \right| \geq t \right\} , \tag{1}$$

where $\mathcal{A}$ denotes the set of all yeast genes, and $\left|\hat{\omega}^*_{ij}\right|$ is the absolute value of the "shrinked" estimate of the partial correlation between genes i and j (i.e., the absolute value of the (i, j)-th entry of matrix $\hat{\Omega}^*$; recall Sect. 3.4). Hence, $\mathcal{I}_t$ is composed of the genes (not in $\mathcal{C}$) for which the partial correlation with at least one gene of $\mathcal{C}$ is greater (in absolute value) than the threshold t. In other words, genes which are dependent (i.e., not independent) of at least one gene in $\mathcal{C}$ (given the remaining genes in the genome) are inferred as NCR-sensitive. Inversely, genes which are independent of all genes in $\mathcal{C}$ are not included in $\mathcal{I}_t$.

[2] The data matrix X is available from `http://rsat.scmbb.ulb.ac.be/~jvanheld/NCR_regulation_2006/gata_boxes_techreport.html`

Concerning the threshold t, we vary its value and use measures of performance related to receiver operator characteristic (ROC) curves (see Sect. 5.1). Of course, the number of inferred genes diminishes with increasing threshold values.

The proposed method can be seen as a (supervised) two-class classifier. For a given threshold t, the genes in $\mathcal{I}_t$ are classified as "positive", i.e., inferred as NCR-sensitive, and the genes not in $\mathcal{I}_t$ (i.e., genes in $\mathcal{A} \setminus (\mathcal{C} \cup \mathcal{I}_t)$) are classified as "negative".

Note that the partial correlation matrix $\hat{\Omega}^*$ only depends on the input data matrix X. Hence, the $\hat{\Omega}^*$ only needs to be computed once when considering different "core" sets and/or different threshold values as long as the same data set is used.

In terms of performance, executing the proposed method (i.e., inferring the partial correlation matrix and building the set $\mathcal{I}_t$) using the statistical software R requires less than 1 minute of CPU time on a 2.2 GHz Intel Core 2 Duo laptop with 2 GB RAM running Mac OS X.

5 Experimental Setup

Our approach requires a set $\mathcal{C}$ of "core" genes to be defined (see Sect. 4.3). Given the available data (see Sect. 2), we use the set of known regulatory genes, $\mathcal{C} = \text{RNCR}$, in Sect. 5.2, and the set of annotated NCR genes, $\mathcal{C} = \text{ANCR}$, in Sect. 5.3. In both cases, positive and negative validation sets, denoted by $\mathcal{P}$ and $\mathcal{N}$, respectively, are defined to assess the predictive power of our approach using the performance measure presented in Sect. 5.1. Note that we perform negative controls to evaluate the significancy of our results in Sect. 5.4. Eventually, we present the procedure for the "final" predictions in Sect. 5.5.

5.1 Performance Measure

As explained in Sect. 4.3, our method can be seen as a two-class classifier. Hence, we use the area under the receiver operator characteristic (ROC) curve (AUC) as performance measure.

A ROC curve is a graphical plot of the true positive rate (TPR) versus the false positive rate (FPR) for different values of the threshold. These two quantities are defined, respectively, as $\text{TPR} = \text{TP}/(\text{TP} + \text{FN})$ and $\text{FPR} = \text{FP}/(\text{FP} + \text{TN})$, where TP, FP, TN, FN represent the numbers of true and false positives, and true and false negatives, respectively. The use of ROC curves is recommended when evaluating binary decision problems in order to avoid effects related to the chosen threshold [21,22].

The AUC reduces ROC performance to a single scalar value representing expected performance [22]. It corresponds to the "probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance" [22]. Note that a random classifier produces the diagonal line between $(0, 0)$ and $(1, 1)$, hence achieving an AUC of 0.5.

However, our interest does not lie in the entire range of FPRs, but rather on very low false positive rates such as FPR < 0.05. We therefore also compute the partial area under the ROC curve (pAUC) [23,24], which is a summary measure of the ROC curve used to make statistical inference when only a region of the ROC space is of interest. It is defined as the area between two FPRs. Here, we focus on FPR $\in (0, u]$ with $u = 0.05$ and denote the corresponding area by pAUC_u. Because the magnitude of pAUC_u depends on u, it is standardized by dividing it by u [24].

Since (p)AUC measures are sample-based estimates, we report jackknife estimates of standard deviations [25] to be able to compare the (p)AUC values [22].

5.2 Known Regulators

We assess the ability of our approach to recover the annotated NCR genes (ANCR) from the set RNCR of known regulators. We thus take $\mathcal{C} = \text{RNCR}$ and $\mathcal{P} = \text{ANCR}$. Concerning the negative validation set $\mathcal{N}$, we consider two alternatives. As already mentioned, we can reasonably assume most of the $\sim 6\,000$ yeast genes not to be targets of NCR. Hence, we first take $\mathcal{N} = \mathcal{A} \setminus \{\mathcal{C} \cup \mathcal{P}\}$ (recall that $\mathcal{A}$ denotes the set of all yeast genes). Next, we consider the set of 89 manually-selected genes known to be insensitive to NCR, i.e., $\mathcal{N} = \text{NNCR}$.

5.3 Annotated NCR Genes

We now run our method with $\mathcal{C} = \text{ANCR}$. The inferred genes are first validated with $\mathcal{P} = \text{ANCR}$ through a leave-one-out procedure. Next, we use the gene sets $\mathcal{G}$, $\mathcal{S}$ and $\mathcal{B}$ steeming from the aforementioned experimental studies [1,2,14].

Leave-One-Out. The ANCR genes form a biologically meaningful set since they are all targets of NCR. Hence, we can expect that any given gene $i \in \text{ANCR}$ is strongly correlated (in terms of partial correlation) to at least one other gene in ANCR. If not, this would imply that gene i interacts indirectly (i.e., through other genes) with the other ANCR genes (by definition of partial correlation) and would be in contradiction with the hypothesis that the ANCR genes form a biologically coherent set.

If our approach to inferring NCR genes is sound, then for each gene $i \in \text{ANCR}$ the maximal partial correlation (in absolute value) of gene i with a gene in $\text{ANCR} \setminus \{i\}$, $\omega_i^{max}(\text{ANCR}) = \max_{j \in \text{ANCR} \setminus \{i\}} \left|\hat{\omega}_{ij}^*\right|$ should be high, relative to the same quantity computed for all genes k not in ANCR ($k \in \mathcal{A} \setminus \text{ANCR}$): $\omega_k^{max}(\text{ANCR}) = \max_{j \in \text{ANCR}} \left|\hat{\omega}_{kj}^*\right|$.

To assess the usefulness of our approach, we thus estimate the p-value p_i, which represents the probability of randomly obtaining a score at least as high as ω_i^{max}, for all $i \in \text{ANCR}$, by the empirical p-value:

$$\hat{p}_i = \frac{\text{Card}\left(\{k \in \mathcal{A} \setminus \text{ANCR} : \omega_k^{max}(\text{ANCR}) \geq \omega_i^{max}(\text{ANCR})\}\right)}{\text{Card}\left(\mathcal{A} \setminus \text{ANCR}\right)}, \tag{2}$$

where $\text{Card}(\mathcal{Z})$ denotes the cardinality of set $\mathcal{Z}$.

Experimental Studies. We also use the genes identified in the aforementioned experimental studies [1,2,14] to validate our approach. Of course, these genes are only putative NCR genes and the three sets identified only partially overlap. Still, in absence of any other validation data, we use these sets to complement the leave-one-out validation procedure described previously. More specifically, we consider all possible combinations of intersections and unions of these three sets (see Sect. 6).

5.4 Negative Control

We perform negative controls to determine whether the results are significant or not. More specifically, for the experiments described in Sects. 5.2 and 5.3 (Paragraph *Experimental Studies*), respectively, we run our method with 1 000 randomly chosen "core" sets $\mathcal{C}_r^i \subset \mathcal{A}$ of cardinality $\text{Card}\left(\mathcal{C}_r^i\right) = \text{Card}\left(\mathcal{C}\right)$, $i = 1, \ldots, 1\,000$. We then perform the validation procedure as described in Sects. 5.2 and 5.3, respectively, and report the mean and standard deviation of the AUC values obtained.

5.5 Final Predictions

Finally, we consider as "core" set $\mathcal{C}$ the known NCR regulators (RNCR) and the annotated NCR genes (ANCR), i.e., $\mathcal{C} = \text{RNCR} \cup \text{ANCR}$, to predict the genes' NCR-sensitivity. Specifically, we compute for each gene $j \in \mathcal{A}$ its maximal partial correlation (in absolute value), denoted by ω_j^{max}, with a gene in the core set (except with itself), $\mathcal{C} \setminus \{j\}$, formally:

$$\omega_j^{max} = \omega_j^{max}\left(\text{RNCR} \cup \text{ANCR}\right) = \max_{i \in \mathcal{C} \setminus \{j\}} \left|\hat{\omega}_{ij}^*\right|, \quad \forall j \in \mathcal{C}. \tag{3}$$

This ω_j^{max} quantity is the "inferred NCR-sensitivity" of gene j.

6 Results and Discussion

Due to space restrictions, we only show the AUC values and have omitted the $\text{pAUC}_{0.05}$ values (see Sect. 5.1). However, the conclusions drawn from the AUC values in this section also apply to the $\text{pAUC}_{0.05}$ values.

6.1 Known Regulators

Table 1 presents the results for $\mathcal{C} = \text{RNCR}$, both for the GGM and the independence graph (see Sect. 5.2). First, we note that all results are significant given the AUC values obtained in the negative control cases. Next, we note the relatively high values (> 0.9) for the GGM compared to the independence graph, which demonstrate the ability of the proposed method to successfully recover the annotated NCR genes from the four known NCR regulators. This suggests that our method is able to identify genes relevant to NCR more efficiently than with the independence graph.

Table 1. AUC values (and jackknife estimates of standard deviations) for $\mathcal{C} = \text{RNCR}$ and $\mathcal{P} = \text{ANCR}$ are given in the "AUC" columns. The first row corresponds to $\mathcal{N} = \mathcal{A} \setminus \{\mathcal{C} \cup \mathcal{P}\}$ and the second one to $\mathcal{N} = \text{NNCR}$. The "negative control" column shows the mean and standard deviation of the AUC over 1 000 repetitions.

	GGM		Independence graph	
Negative set $\mathcal{N}$	AUC	Neg. control	AUC	Neg. control
$\mathcal{A} \setminus \{\mathcal{C} \cup \mathcal{P}\}$	0.910 (0.034)	0.501 (0.049)	0.678 (0.11)	0.499 (0.049)
NNCR	0.909 (0.039)	0.501 (0.059)	0.668 (0.14)	0.501 (0.059)

6.2 Annotated NCR Genes

Results for $\mathcal{C} = \text{ANCR}$ and $\mathcal{N} = \mathcal{A} \setminus \{\mathcal{C} \cup \mathcal{P}\}$ are shown in Table 2 (see Paragraph *Experimental Studies* in Sect. 5.3). Results for $\mathcal{C} = \text{ANCR}$ and $\mathcal{N} = \text{NNCR}$ are similar to those in Table 2 and have been omitted due to space restrictions. As previously, we note that all results are significant given the AUC values obtained in the negative control cases. We also note that the best AUC values are obtained for sets $\mathcal{P}$ that contain genes found in at least two of the aforementioned studies [1,2,14] (i.e., for intersections of at least two of the sets $\mathcal{G}$, $\mathcal{S}$ and $\mathcal{B}$). In other words, the stronger is the "consensus" on the NCR-sensitivity of a gene, the higher is the probability of this gene to be identified as such by our approach. However, we note that, in this case, the GGM only slightly outperforms the independence graph.

Table 2. AUC values (and jackknife estimates of standard deviations) for $\mathcal{C} = \text{ANCR}$ and $\mathcal{N} = \mathcal{A} \setminus \{\mathcal{C} \cup \mathcal{P}\}$ are given in the "AUC" columns. For $\mathcal{P}$, we consider all possible combinations (in terms of unions and intersections) of the sets $\mathcal{G}$, $\mathcal{S}$ and $\mathcal{B}$ (from which we remove genes in $\mathcal{C}$). The "negative control" column shows the mean and standard deviation of the AUC over 1 000 repetitions.

	GGM		Independence graph	
$\mathcal{P}$	AUC	Neg. control	AUC	Neg. control
$\mathcal{G} \setminus \mathcal{C}$	0.696 (0.021)	0.499 (0.028)	0.638 (0.028)	0.499 (0.027)
$\mathcal{S} \setminus \mathcal{C}$	0.709 (0.030)	0.499 (0.039)	0.650 (0.052)	0.500 (0.040)
$\mathcal{B} \setminus \mathcal{C}$	0.656 (0.034)	0.498 (0.038)	0.637 (0.049)	0.501 (0.036)
$\{\mathcal{G} \cup \mathcal{S}\} \setminus \mathcal{C}$	0.690 (0.018)	0.501 (0.024)	0.635 (0.026)	0.501 (0.025)
$\{\mathcal{G} \cup \mathcal{B}\} \setminus \mathcal{C}$	0.671 (0.017)	0.500 (0.023)	0.628 (0.026)	0.502 (0.023)
$\{\mathcal{S} \cup \mathcal{B}\} \setminus \mathcal{C}$	0.671 (0.026)	0.501 (0.028)	0.634 (0.036)	0.500 (0.027)
$\{\mathcal{G} \cup \mathcal{S} \cup \mathcal{B}\} \setminus \mathcal{C}$	0.669 (0.017)	0.500 (0.021)	0.624 (0.024)	0.500 (0.021)
$\{\mathcal{G} \cap \mathcal{S}\} \setminus \mathcal{C}$	0.792 (0.053)	0.500 (0.064)	0.703 (0.071)	0.499 (0.069)
$\{\mathcal{G} \cap \mathcal{B}\} \setminus \mathcal{C}$	0.836 (0.034)	0.501 (0.085)	0.779 (0.078)	0.498 (0.084)
$\{\mathcal{S} \cap \mathcal{B}\} \setminus \mathcal{C}$	0.867 (0.032)	0.499 (0.107)	0.805 (0.073)	0.499 (0.109)
$\{\mathcal{G} \cap \mathcal{S} \cap \mathcal{B}\} \setminus \mathcal{C}$	0.855 (0.047)	0.493 (0.144)	0.736 (0.103)	0.503 (0.145)

Concerning the leave-one-out procedure (see Paragraph *Leave-One-Out* in Sect. 5.3), 22, 24 and 31 out of 37 ANCR genes are significantly identified as such, with P-val. ≤ 0.05, P-val. ≤ 0.10 and P-val. ≤ 0.15, respectively. This is to be compared with the 26, 28, 17 and 16 ANCR genes identified as such in the aforementioned experimental studies [1,2,14] and by the independence graph (P-val. ≤ 0.10), respectively.

6.3 Final Predictions

We ranked all genes by decreasing "inferred NCR-sensitivity" (see Sect. 5.5). Out of the 38 genes which are NCR regulators (RNCR) and/or targets of NCR (ANCR), 13, 22 and 24 are in the top 50, 100, and 600 genes ($\sim 10\%$ of all genes) of the ranking (omitted due to space restrictions), respectively. Among the 14 genes not appearing in the top 600 genes, one (PUT4) is known to be a "difficult case" because its two GATA-boxes are non-canonical [1].

Further, using the *feature-map* program from RSAT [20], we note that the 24 RNCR and/or ANCR genes appearing in the top 600 genes have a high density of GATA boxes in their upstream noncoding sequences, while the remaining 14 genes have a comparatively lower density (figures not shown because of space restrictions).

Finally, 4, 9 and 12 of the 16 genes identified in each of the three aforementioned studies [1,2,14] (i.e., the intersection of the sets $\mathcal{G}$, $\mathcal{S}$ and $\mathcal{B}$) appear in the top 50, 100, and 600 genes of the ranking, respectively. In other words, putative genes having the largest "consensus" are relatively well ranked by our approach.

7 Conclusion

We proposed an approach based on Gaussian graphical models (GGMs) to identify putative NCR genes from putative NCR regulatory motifs and over-represented motifs in the upstream noncoding sequences of annotated NCR genes. Because of the high-dimensionality of the data, we used a shrinkage estimator of the covariance matrix to infer the GGMs, which is statistically efficient and fast to compute.

We showed that our approach made significant and biologically valid predictions by comparing these predictions to annotated and putative NCR genes, and by performing negative controls. We also showed that the GGM is more effective than the independence graph. These results suggest that our approach can successfully identify potential NCR genes in *S. cerevisiae*.

Preliminary results obtained by applying the proposed approach to the *Saccharomyces*, *Saccharomycetaceae*, *Saccharomycetes*, *Ascomycota* and *Fungi* taxa suggest that the annotated NCR genes of *S. cerevisiae* also form a biologically coherent set in each of these taxa. Future work will extend the counting of motifs in upstream nonconding sequences of orthologs of *S. cerevisiae* genes for these taxonomical levels to improve the prediction of NCR genes.

Note that the proposed approach can readily be adapted to any type of data (e.g., expression data), and to any biological process of interest in any sequenced

organism. Finally, note that we do not endorse the GGM as the "true model" of multivariate dependencies between genes. Rather, we see it as a useful exploratory tool.

References

1. Godard, P., Urrestarazu, A., Vissers, S., Kontos, K., Bontempi, G., van Helden, J., André, B.: Effect of 21 different nitrogen sources on global gene expression in the yeast *Saccharomyces cerevisiae*. Molecular and Cellular Biology 27, 3065–3086 (2007)
2. Scherens, B., Feller, A., Vierendeels, F., Messenguy, F., Dubois, E.: Identification of direct and indirect targets of the Gln3 and Gat1 activators by transcriptional profiling in response to nitrogen availability in the short and long term. FEMS Yeast Research 6, 777–791 (2006)
3. Kontos, K., Godard, P., André, B., van Helden, J., Bontempi, G.: Machine learning techniques to identify putative genes involved in nitrogen catabolite repression in the yeast *Saccharomyces cerevisiae*. BMC Proceedings 2, S5 (2008)
4. Lauritzen, S.L.: Graphical Models. Oxford Statistical Science Series. Clarendon Press, Oxford (1996)
5. Simonis, N., Wodak, S.J., Cohen, G.N., van Helden, J.: Combining pattern discovery and discriminant analysis to predict gene co-regulation. Bioinformatics 20, 2370–2379 (2004)
6. Schäfer, J., Strimmer, K.: A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. Statistical Applications in Genetics and Molecular Biology 4, 32 (2005)
7. Dobra, A., Hans, C., Jones, B., Nevins, J., Yao, G., West, M.: Sparse graphical models for exploring gene expression data. Journal of Multivariate Analysis 90, 196–212 (2004)
8. Castelo, R., Roverato, A.: A robust procedure for Gaussian graphical model search from microarray data with p larger than n. Journal of Machine Learning Research 7, 2621–2650 (2006)
9. Magwene, P., Kim, J.: Estimating genomic coexpression networks using first-order conditional independence. Genome Biology 5, R100 (2004)
10. Wille, A., Zimmermann, P., Vranová, E., Fürholz, A., Laule, O., Bleuler, S., Hennig, L., Prelić, A., von Rohr, P., Thiele, L., Zitzler, E., Gruissem, W., Bühlmann, P.: Sparse graphical Gaussian modeling of the isoprenoid gene network in *Arabidopsis thaliana*. Genome Biology 5, R92 (2004)
11. Kontos, K., Bontempi, G.: Nested q-partial graphs for genetic network inference from "small n, large p" microarray data. In: Elloumi, M., Küng, J., Linial, M., Murphy, R., Schneider, K., Toma, C. (eds.) BIRD 2008. CCIS 13, pp. 273–287. Springer, Heidelberg (2008)
12. Ledoit, O., Wolf, M.: A well-conditioned estimator for large-dimensional covariance matrices. Journal of Multivariate Analysis 88, 365–411 (2004)
13. Cooper, T.G.: Transmitting the signal of excess nitrogen in *Saccharomyces cerevisiae* from the Tor proteins to the GATA factors: connecting the dots. FEMS Microbiology Reviews 26, 223–238 (2002)
14. Bar-Joseph, Z., Gerber, G., Lee, T., Rinaldi, N., Yoo, J., Robert, F., Gordon, D., Fraenkel, E., Jaakkola, T., Young, R., et al.: Computational discovery of gene modules and regulatory networks. Nature Biotechnology 21, 1337–1342 (2003)

15. Butte, A., Tamayo, P., Slonim, D., Golub, T., Kohane, I.: Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. Proceedings of the National Academy of Sciences 97, 12182–12186 (2000)
16. Whittaker, J.: Graphical Models in Applied Multivariate Statistics. John Wiley and Sons, Inc., Chichester (1990)
17. Edwards, D.: Introduction to Graphical Modelling, 2nd edn. Springer Texts in Statistics. Springer, Heidelberg (2000)
18. Schäfer, J., Strimmer, K.: An empirical Bayes approach to inferring large-scale gene association networks. Bioinformatics 21, 754–764 (2005)
19. Dykstra, R.: Establishing the positive definiteness of the sample covariance matrix. The Annals of Mathematical Statistics 41, 2153–2154 (1970)
20. van Helden, J.: Regulatory sequence analysis tools. Nucleic Acids Research 31, 3593–3596 (2003)
21. Provost, F., Fawcett, T., Kohavi, R.: The case against accuracy estimation for comparing induction algorithms. In: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 445–453. Morgan Kaufmann, San Francisco (1998)
22. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters 27, 861–874 (2006)
23. McClish, R.J.: Analyzing a portion of the ROC curve. Medical Decision Making 9, 190–195 (1989)
24. Jiang, Y.L., Metz, C.E., Nishikawa, R.M.: A receiver operating characteristic partial area index for highly sensitive diagnostic tests. Radiology 201, 745–750 (1996)
25. Efron, B.: Nonparametric estimates of standard error: the jackknife, the bootstrap and other methods. Biometrika 68, 589–599 (1981)

Chronic Rat Toxicity Prediction of Chemical Compounds Using Kernel Machines

Georg Hinselmann, Andreas Jahn, Nikolas Fechner, and Andreas Zell

Wilhelm-Schickard-Institute for Computer Science, Dept. Computer Architecture, University of Tübingen, Tübingen, Germany

Abstract. A recently published study showed the feasibility of chronic rat toxicity prediction, an important task to reduce the number of animal experiments using the knowledge of previous experiments. We benchmarked various kernel learning approaches for the prediction of chronic toxicity on a set of 565 chemical compounds, labeled with the Lowest Observed Adverse Effect Level, and achieved a prediction error close to the interlaboratory reproducibility. ϵ-Support Vector Regression was used in combination with numerical molecular descriptors and the Radial Basis Function Kernel, as well as with graph kernels for molecular graphs, to train the models. The results show that a kernel approach improves the Mean Squared Error and the Squared Correlation Coefficient using leave-one-out cross-validation and a seeded 10-fold-cross-validation averaged over 10 runs. Compared to the state-of-the-art, the Mean Squared Error was improved up to $\mathrm{MSE}_{\mathrm{loo}}$ of 0.45 and $\mathrm{MSE}_{\mathrm{cv}}$ of 0.46 ± 0.09 which is close to the theoretical limit of the estimated interlaboratory reproducibility of 0.41. The Squared Empirical Correlation Coefficient was improved to $\mathrm{Q}^2_{\mathrm{loo}}$ of 0.58 and $\mathrm{Q}^2_{\mathrm{cv}}$ of 0.57 ± 0.10. The results show that numerical kernels and graph kernels are both suited for predicting chronic rat toxicity for unlabeled compounds.

1 Introduction

Chronic toxicity prediction (the prediction of toxic long-term effects of some drug) is a fundamental problem in pharmaceutical and food industry. The difficulty of this problem arises from the different types of toxicity (e.g. renal, hepatic or gastrointestinal). The experimental toxicity is obtained by expensive and ethically problematic animal experiments. Therefore, a need for in silico prediction systems exists to filter out toxic compounds based on the knowledge of previous experiments [3,25].

Several studies attempting to predict chronic toxicity have been published [12,13,20,22,23]. Recently, Mazzatorta et al. [11] published a model based on linear regression using two-dimensional molecular descriptors, selected by a Genetic Algorithm, to predict the Lowest Observed Adverse Effect Level (LOAEL). The LOAEL is a convenient measure to indicate long-term toxic effects. A dataset with 567 chemical compounds was compiled from various studies containing

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 25–36, 2009.

chemical compounds and their LOAEL values for chronic rat toxicity. The proposed model uses 20 two-dimensional descriptors computed by the software dragon 5.4 and has a MSE_{loo} of 0.53 and a Q^2_{loo} of 0.5.

To the best of our knowledge, our paper is the first attempt using kernel machines to predict chronic toxicity. We extend the experimental setup to three-dimensional methods and make use of graph kernels. A recent development in machine learning in cheminformatics is the use of graph kernels, i.e. pairwise symmetric, positive semidefinite similarities on molecular graphs [4,8,10,14,15,16]. Graph kernel based methods can learn on molecular graphs without explicitly defining features.

We applied ϵ-Support Vector Regression in combination with different kernel functions to predict the LOAEL. Support Vector Machines solve learning tasks by mapping input data into a high-dimensional feature space. This projection is not conducted explicitly, but implicitly by a similarity preserving embedding. A regularization term is included in the formulation ensuring the complexity of the generated model, i.e. the tendency of over-fitting is penalized. This usually results in a good generalization performance.

The results on the benchmark set of Mazzatorta et al. [11] indicate the kernel approaches are at least comparable in most cases. The MSE_{loo} of competing methods ranges from 0.45 to 0.51 with a Q^2_{loo} between 0.52 and 0.58. Additionally, we benchmarked the methods with a seeded 10-fold cross-validation averaged over 10 runs with comparable results.

We reduced the MSE_{loo} of the chronic rat toxicity problem presented in [11] with an expected prediction quality close to the MSE of the interlaboratory reproducibility of 0.41. Q^2_{loo} was also improved from 0.50 to 0.58. The resulting models are expected to have good generalization capabilities because no feature selection was used, the models have far less variables and use non-linear mappings.

2 Materials and Methods

2.1 Chronic Toxicity Data Set

The original data set published by Mazzatorta et al. [11] is a compilation of 567 chemical compounds, each labeled with a LOAEL value. The LOAEL is a measure for the minimum dose of a drug in mg per body weight (in kg) per day where a toxic response is observable.

To compile a consistent data set, Mazzatorta et al. [11] chose a subset from different sources [13,23,22] which contains compounds for oral chronic toxicity studies for *Rattus norvegicus* that were observed at least for more than 180 days. Due to the different sources of the data, a squared interlaboratory reproducibility of 0.41 was estimated using samples with at least two valid LOEAL values [11].

We converted the canonical SMILES strings of the original data into three-dimensional structures using CORINA3D [6]. Two structures with a corrupted SMILES string representation (ID 299 and 997) were removed from the data set. The LOAEL value was transformed to a logarithmic ($\log_{10}$) scale.

2.2 Support Vector Regression

A supervised machine learning algorithm infers a model by learning a set of labeled samples. In the case of chemical compounds, for example, a sample is labeled with its activity against a specific target, a physicochemical property or, in this case, a toxicity level. The aim is to train a model with good predictive performance on unknown instances.

Machine learning techniques have been successfully applied in cheminformatics and Support Vector Machines (SVMs) are among the most important methods in this field. A suitable source of information on SVMs is the book of Schölkopf and Smola [17]. SVMs use a kernel function k which has to be symmetric and positive semi-definite for similarity-based learning. The ϵ-Support Vector Regression (ϵ-SVR) problem is formulated as the minimization of the following functional:

$$H(f) = \sum_{i=1}^{m} \frac{C}{m} |y_i - f(\mathbf{x_i})|_\varepsilon + \underbrace{\frac{1}{2} \|f\|_k^2}_{\text{complexity penalty}}$$

where $|y_i - f(\mathbf{x_i})|_\varepsilon = \max\{0, |y_i - f(\mathbf{x_i})| - \varepsilon\}$ (the ϵ-insensitive loss). $\|f\|_k^2$ is the norm of f in some Hilbert Space. The constant C can be regarded as a penalty term. High values for C indicate a potential overfitting, small values could indicate underfitting. Only samples outside a tube of 2ϵ around the regression function contribute to the cost function. The parameters ϵ and C have to be optimized for each problem. y_i is the label of the ith sample $\mathbf{x_i}$, m is the size of the training set. The solution $f(\mathbf{x})$, i.e. the regression function, of the Quadratic Programming (QP) dual formulation is

$$f(\mathbf{x}) = \sum_{i=1}^{m} \underbrace{(\alpha_i - \alpha_i^*)}_{\text{weight}} k(x, x_i) + \underbrace{\beta_0}_{\text{bias}}$$

where $\alpha_i, \alpha_i^* \in [0, C]$ are the Lagrange multipliers, k is the kernel function and β_0 the bias. $\sum_{i=1}^{m} (\alpha_i - \alpha_i^*) = 0$, so the sum over all weights has to be zero. A support vector has a non-zero weight. Hence, the prediction of the label of an unknown sample is computed by its weighted similarity to the set of support vectors.

2.3 Kernels for Molecular Graphs

This section gives an overview of the kernel functions that were used in the experimental section. We start with basic numerical kernels, followed by topology-based approaches and end up with graph kernels deriving information from three-dimensional coordinates.

Radial Basis Function Kernel. The Radial Basis Function (RBF) Kernel is suitable for the pairwise comparison of two samples $\mathbf{x_i}, \mathbf{x_j}$ with a vectorial representation. It is defined as $k_{RBF}(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\frac{\|\mathbf{x_i}-\mathbf{x_j}\|^2}{2\sigma^2}\right)$, the σ parameter gives the standard deviation of the Gaussian. The descriptors for this study were computed with dragonX 1.4 [19]. Most of the descriptors are delineated in the Handbook of Molecular Descriptors [21]. Since version 1.4 of dragonX, two descriptor blocks $(780 \cdot 2)$ have been added, describing atom pairs with a binned topological occurrence and their corresponding frequency. Thus, a dragonX 1.4 descriptor vector is a combination of classical descriptors and an atom pair fingerprint. The software provides about 3200 molecular descriptors in total divided into 22 descriptor blocks.

The descriptors can be divided into four classes. 0D descriptors describe simple counts like atom types or bonds. 1D descriptors are fragment counts, 2D descriptors are topology-related. 3D descriptors are based on geometrical coordinates and therefore require a valid conformation. We chose four descriptor configurations for the RBF Kernel: (1) RBF-ALLBLOCKS used the complete set of descriptors, (2) RBF-DESCONLY used all descriptors except the atom pair information, (3) RBF-2DDESC used all blocks of descriptors that do not need three-dimensional coordinates, (4) RBF-APFP used the two blocks of atom pair information only. Hydrogens were included for the computation. Each descriptor $\mathbf{d_i}$ was standardized by $d_{ij} \leftarrow \frac{d_{ij}-\mu}{\sigma}$, where μ denotes the mean of the ith descriptor of j molecules and σ its standard deviation. All constant attributes were filtered out and any feature with a missing value was excluded from the kernel computation.

Marginalized Graph Kernel. The Marginalized Graph Kernel (MARG) determines the similarity of two molecular graphs by means of common labeled random walks. Thus, it is defined as the expectation of a kernel of all pairs of walks of two graphs [8].

Tanimoto Kernel on Depth-First Search Fingerprints. The Tanimoto Kernel compares arbitrary nominal feature sets $F_i = \{f_{i1}, f_{i2}, \cdots, f_{im}\}$ and $F_j = \{f_{j1}, f_{j2}, \cdots, f_{jn}\}$. Ralaivola et al. [14] introduced the Tanimoto Kernel and the MinMax Kernel for the prediction of chemical properties. A further study was carried out by Azencott et al. [1]. The Tanimoto Kernel is a valid kernel function [7,14]. It is defined as $k_{TM}(F_i, F_j) = \frac{|F_i \cap F_j|}{|F_i \cup F_j|}$.

The Tanimoto Kernel compares the distance of two feature sets in their joint space, without defining this space explicitly. An useful property is that new features lead to an increased dissimilarity. The features that are not contained in both structures are simply omitted. Intuitively, this representation corresponds to a fingerprint of unlimited size.

In this study, the Tanimoto Kernel was used on paths obtained from a depth-first search (DFS) with depth d from all atoms. In our implementation the fingerprint algorithm from the Chemistry Development Kit [18] was modified, so that the complete list of canonical paths of depth $d \in \mathbb{N}$ is returned.

Optimal Assignment Kernel. The idea of the Optimal Assignment Kernel (OAK) is to compute an optimal weighted assignment on two sets of objects and to use the resulting similarity as kernel function. The underlying problem is to solve a matching on a complete bipartite graph with respect to $\max w(M) = \max \sum_{e \in M} w(e)$, where $w(M)$ is the sum of the weights of the matching edges $e(i,j)$, between two objects i, j of two disjoint sets. Each feature of the smaller set has to be assigned to exactly one feature of the other set. The OAK was introduced by Fröhlich et al. and successfully applied to attributed molecular graphs [5]. The kernel function of the optimal weighted assignment is defined as follows:

$$k_{OA}(\mathbf{x}, \mathbf{x}') := \begin{cases} \max_{\pi \in \Pi(\mathbf{x}')} \sum_{i=1}^{|\mathbf{x}|} \kappa\left(x_i, x'_{\pi(i)}\right), \text{ if } |\mathbf{x}'| > |\mathbf{x}| \\ \max_{\pi \in \Pi(\mathbf{x})} \sum_{j=1}^{|\mathbf{x}'|} \kappa\left(x_{\pi(j)}, x'_j\right), \text{ otherwise} \end{cases}$$

where $\mathbf{x} := (x_1, x_2, \ldots, x_{|x|})$ and $\mathbf{x}' := (x'_1, x'_2, \ldots, x'_{|x'|})$ are the sets of atoms which compose the corresponding molecular graph. $\Pi(\mathbf{x})$ denotes all possible permutations of $\mathbf{x}$ and $\Pi(\mathbf{x}')$ of $\mathbf{x}'$, respectively. The atom-wise similarity is determined by κ which can be any suitable kernel function on atomic attributes. The OAK uses a local atom environment which encodes the molecular neighborhood up to a defined depth using nominal and numerical features. $k_{OA}(\mathbf{x}, \mathbf{x}')$ computes the maximum score of all possible permutations.

Optimal Assignment Kernels are pseudo-kernels [24]. Therefore, each kernel matrix has to be fixed by the transformation $K \leftarrow K - \lambda_{min} I$, where λ_{min} represents the lowest negative eigenvalue of the kernel matrix K and I the identity matrix. A closely related variant of the OAK is the Optimal Assignment Kernel

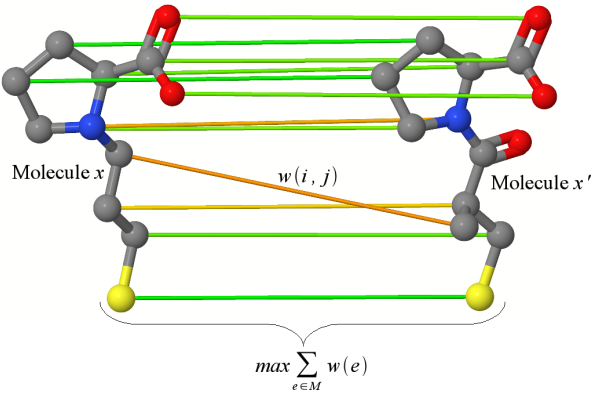
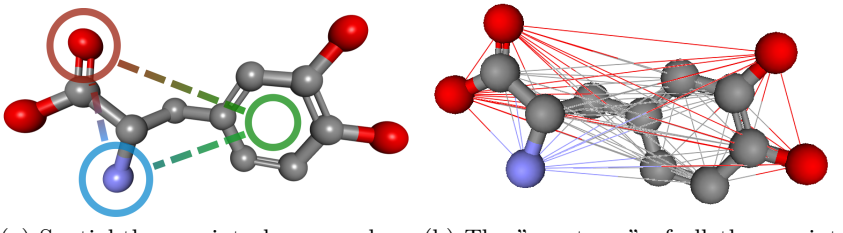


Fig. 1. Example of an optimal assignment between two molecules where the color of the matching edges is determined by its quality (green: optimal matching, red: bad matching)

on Reduced Graphs (OARG) which uses featured subgraphs, for example rings, as vertices instead of the full atom set. This reduces memory requirements and computation time, but decreases the information content of the featured graph.

OAK with Flexibility Extension. This modification of the original OAK (OAK_{flex}, manuscript accepted at the *Journal of Chemical Information and Modeling*, 2009) incorporates a flexibility extension into the atom-wise similarity calculation. The flexibility of the local neighborhood of an atom is encoded as a list of flexibility objects. Each object has a fixed number of parameters describing the space in which the neighboring atom can be found. A RBF Kernel calculates the similarity between two flexibility objects using the parameters of the spaces. The overall similarity of the flexibility between two atoms is computed as follows: $k_{flex} = \frac{\sum_{i=1}^{|a|} k_{rot}(a_i, b_{\pi_{rot(i)}})}{\sqrt{|a||b|}}$, w.l.o.g. $|a| < |b|$, where a and b are the lists of the flexibility objects, k_{rot} is the RBF Kernel and $b_{\pi_{rot(i)}}$ is the i-th element of a subset of b so that the sum of all pairwise similarities is maximized. This problem is equal to the optimal assignment problem but due to the computational cost it is solved by a greedy heuristic. The final similarity of the flexibility is added to the respective edge in the bipartite graph of the OAK.

***p*-Point-Pharmacophore Kernel.** From an abstract point of view, a pharmacophore is a three-dimensional relationship between pharmacophore interaction features in a molecule responsible for an interaction with a pharmaceutical target. Pharmacophore Kernels [10] between two molecules m, m' are defined as $k(m, m') = \sum_{i=0}^{k} \sum_{j=0}^{l} \kappa(p_i, p_j)$.



(a) Spatial three-point-pharmacophore (b) The "spectrum" of all three-point relationships

Fig. 2. The Pharmacophore Kernel extracts all spatial p-point relationships of two structures (in this case $p = 3$) and compares the set of all linear patterns efficiently using a tree search

The total similarity is determined by summing up all pairwise kernel similarities between all pharmacophores p_i, p_j in two molecules. The information of a pharmacophore is defined by its distances and pharmacophore features.

Table 1. Overview of applied kernel functions. Possible feature types are: 0D = atom and bond counts, 1D = fragment counts, 2D = topological and related descriptors, 3D = descriptors using geometrical coordinates.

Kernel	Parameters	Features	Type
RBF-ALLBLOCKS	σ	All descriptor blocks	0D to 3D
RBF-2DDESC	σ	All descriptor blocks, except 3D	0D to 2D
RBF-DESCONLY	σ	All descriptors, except atom pair block	0D to 3D
RBF-APFP	σ	Topological atom pairs with frequency	2D
Tanimoto	d	Linear fragments from DFS search	2D
OAK		Local atom environments (LAE)	0D to 2D
OARG		LAE in reduced graph	2D
MARG		Linear fragments (random walks)	2D
OAK_{flex}	x, y	LAE with flexibility information	0D to 3D
Pharmacophore	w	Spatial three-point patterns	3D

Mahé et al. [10] proposed a general kernel function of the form $\kappa(p_i, p_j) = k_I(p_i, p_j) \times k_S(p_i, p_j)$, where the intrinsic kernel k_I provides a similarity measure for two pharmacophore features and the spatial kernel k_S a measure for their distance.

In this study, the vertices of the molecular graph are colored by the hash code of the binned Gasteiger-Marsili partial charges plus the atomic symbol. The bonds are labeled by a discrete linear binning function with bin size of w in Ångstrom, following the work of Mahé et al. [10]. Thus, the distances and the atom types can be compared by the Dirac Kernel. For a fast computation it is convenient to apply a p-spectrum kernel-like approach [9] using search trees to compare the p-point-pharmacophores of two structures. The ensemble of all p-point-pharmacophores is compared in linear computation time by a recursive comparison using tries.

2.4 Availability of Source Code, Tools and Datasets

The benchmark data set used in this paper is a supplement of the study published by Mazzatorta et al. [11]. Available kernel implementations are OAK, OARG and MARGK[1]. The source code of the Pharmacophore Kernel and Tanimoto Kernel including all preprocessing steps and the modification of the LIBSVM with various enhancements for rapid benchmarking of kernels can be obtained from the authors upon request. The implementations of the Pharmacophore Kernel and Tanimoto Kernel use parts of the CDK[2] [18], an open source cheminformatics toolbox. The OAK, OARG and MARGK implementations use parts of the JOELib2[3] library. dragonX 1.4 is commercial software. Nonetheless, the descriptor files and the structures with optimized geometry can be obtained from the authors.

[1] http://www.dkfz.de/mga2/people/froehlich/
[2] http://sourceforge.net/projects/cdk
[3] http://joelib.sourceforge.net/

2.5 Experimental Setup

The parameters of the kernel functions were set as follows. The search depth d of the Tanimoto Kernel was optimized in $\{2, 3, \cdots, 12\}$, the bin width w of the distance labels of the Pharmacophore Kernel in $\{0.25, 0.50, \cdots, 5.75\}$, the σ parameter of the RBF Kernel in $\{1.0, 2.0, \cdots, 9.0\}$. OAK_{flex} uses two parameters x and y determining the influence of the flexibility similarity on the OAK. They were both optimized in $\{0.00, 0.05, \cdots, 1.00\}$. The remaining kernels were used with their default parametrization.

For each learning problem, defined by the kernel matrix and the vector of labels, the relevant ϵ-SVR parameters were optimized by model selection. The parameter grid was set to $log_2 C \in \{-4, -3, \cdots, 10\}$ and $log_2 \epsilon \in \{-8, -7, \cdots, -1\}$.

To learn and validate the models and to compare the different methods, a modified version of LIBSVM 2.85 [2] was applied. All parameter combinations were evaluated by a 10-fold-cross-validation with random folds. This process was repeated 10 times for each kernel matrix to avoid overfitting. The folds were generated with a seeded random number generator resulting in equal folds for different kernels on the same problem. Additionally, a leave-one-out cross-validation was applied.

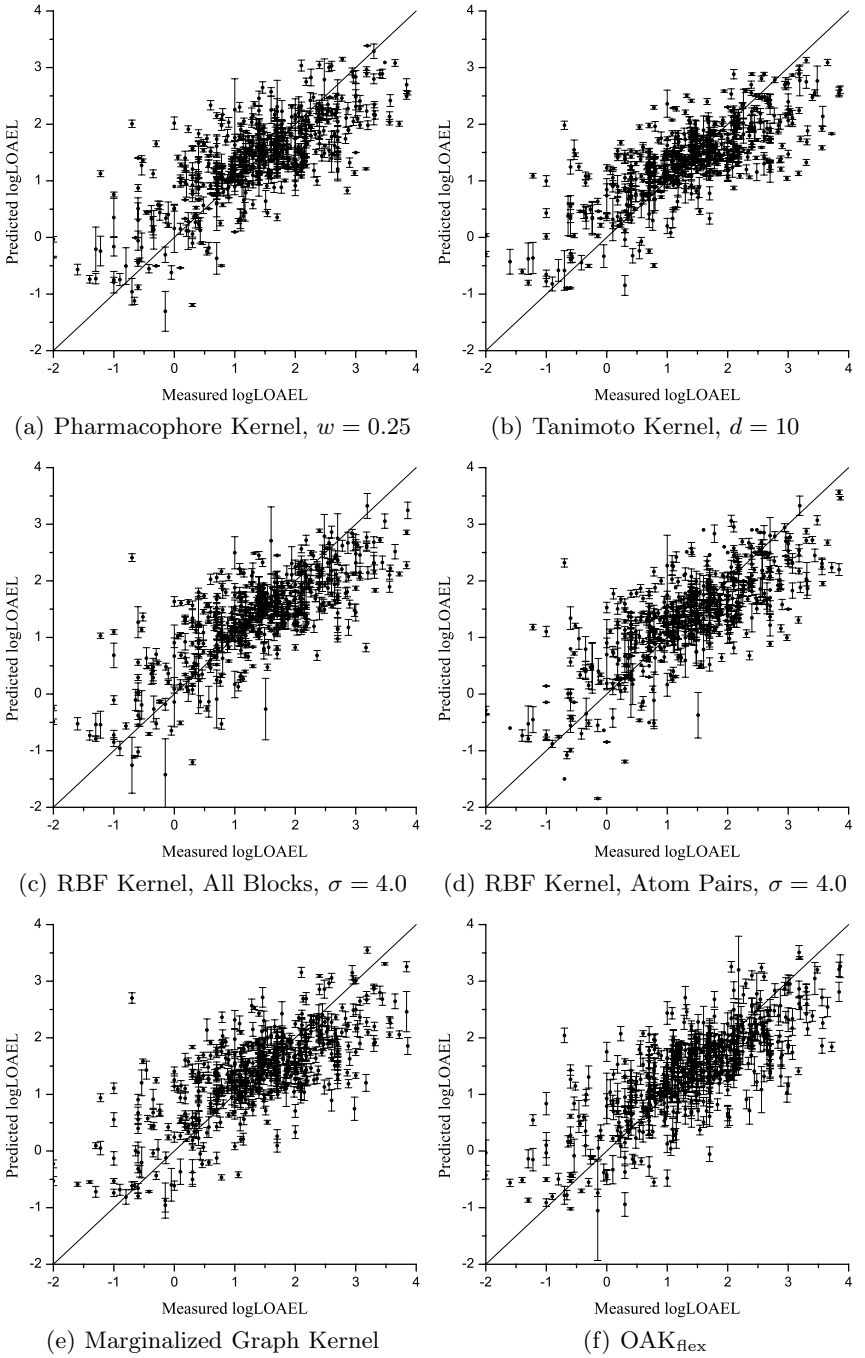
3 Results

Table 2 summarizes the results of the benchmark runs using an averaged 10-fold cross-validation over 10 runs and leave-one-out cross-validation. The leave-one-out cross-validation enables a fair comparison with the literature results. The statistical quality measures are Mean Squared Error (MSE), Averaged Absolute Error (AAE) and Squared Correlation Coefficient (Q^2).

The results are sorted according to the MSE_{cv}. The MSE_{cv} ranges from 0.46 ± 0.09 (RBF-ALLBLOCKS) to 0.60 ± 0.13 (OARG). MSE_{loo} ranges from 0.45 (RBF-ALLBLOCKS, RBF-DESCONLY) to 0.58 (OARG). The Pharmacophore Kernel, OAK_{flex} and Tanimoto Kernel achieve a similar MSE_{loo} ranging

Table 2. Benchmarks obtained by 10 averaged runs of a 10-fold cross-validation and leave-one-out cross-validation

Method	MSE_{cv}	AAE_{cv}	Q^2_{cv}	MSE_{loo}	AAE_{loo}	Q^2_{loo}
RBF-ALLBLOCKS	0.46 ± 0.09	0.51 ± 0.05	0.56 ± 0.09	0.45	0.50	0.56
OAK_{flex}	0.46 ± 0.09	0.52 ± 0.06	0.56 ± 0.09	0.46	0.52	0.58
RBF-DESCONLY	0.46 ± 0.10	0.51 ± 0.06	0.57 ± 0.10	0.45	0.50	0.58
RBF-2DDESC	0.47 ± 0.12	0.51 ± 0.06	0.56 ± 0.10	0.46	0.50	0.56
RBF-APFP	0.48 ± 0.10	0.53 ± 0.05	0.54 ± 0.11	0.47	0.52	0.56
Tanimoto	0.49 ± 0.10	0.53 ± 0.06	0.54 ± 0.09	0.47	0.52	0.55
Pharmacophore	0.49 ± 0.11	0.53 ± 0.06	0.54 ± 0.10	0.48	0.53	0.55
OAK	0.52 ± 0.10	0.56 ± 0.06	0.51 ± 0.11	0.50	0.55	0.52
MARG	0.52 ± 0.11	0.56 ± 0.06	0.51 ± 0.08	0.51	0.55	0.52
OARG	0.60 ± 0.13	0.59 ± 0.07	0.43 ± 0.12	0.58	0.58	0.45



(a) Pharmacophore Kernel, $w = 0.25$

(b) Tanimoto Kernel, $d = 10$

(c) RBF Kernel, All Blocks, $\sigma = 4.0$

(d) RBF Kernel, Atom Pairs, $\sigma = 4.0$

(e) Marginalized Graph Kernel

(f) OAK_{flex}

Fig. 3. Regression plots with error bars regarding the mean and standard deviation of the MSE_{cv} for the prediction of the log LOAEL (on a $\log_{10}$ scale) of each sample

from 0.46 to 0.48. Q^2_{cv} is between 0.43 ± 0.12 (OARG) and 0.56 ± 0.09 (RBF-DESCONLY), the Q^2_{loo} is at least equal.

Figure 3 illustrates the prediction performance. The mean prediction of each sample and its standard deviation is plotted for six different kernels. The plots show that the LOAEL can be predicted for low and large doses. The mean percentage of structures predicted with an AAE larger than one order of magnitude is in the range 12.4% (RBF-2DDESC) and 19.4% (OARG). The best graph kernel is the Tanimoto with 13.5% regarding the AAE_{cv}.

We computed eigenvalue decompositions of all kernel matrices using JAMA[4] to ensure that all kernel matrices are positive semi-definite. All matrices passed this test. A pairwise comparison using the MSE values obtained from the multirun cross-validations in a paired-sample Wilcoxon test revealed that OAK, MARG and OARG were significantly worse than all methods ranked above in Table 2.

4 Discussion and Conclusion

In this paper, the prediction of chronic toxic effects of chemical compounds on rats was addressed using kernel machines. An accurate in silico prediction of the LOAEL helps to reduce the number of animal experiments, cost, time and to increase sustainability.

By using ϵ-SVR and various kernel functions, the prediction error as well as the empirical correlation between prediction and measurements could be improved compared to the state-of-the-art. Numerical kernels and graph kernels are well-suited methods to solve this task. The simplest configuration, the RBF Kernel with 2D descriptors, gives satisfying results. This is in agreement with the work of Mazzatorta et al. [11]. Flexibility information improves the predictive performance of the OAK. This may be due to the impact of the flexibility of a molecule on the bioavailability of a drug. Graph kernels are a good alternative to model the LOAEL. 3D descriptors and the consideration of geometrical pharmacophores do not improve the predictive performance, this is also in agreement with [11].

In general, the MSE_{loo} of 0.53 [11] could be improved to 0.45 and Q^2_{loo} from 0.5 to 0.58 using an RBF Kernel with all descriptors. The RBF Kernel with reduced descriptor blocks or atom pair information only also works well. Models obtained by using the OAK_{flex}, the Tanimoto Kernel or the Pharmacophore Kernel have a comparable quality. The good performance of the these kernels may be due to the consideration of fragment-like patterns which are regarded for the similarity computation. Many compounds turn toxic when they are metabolized revealing the actual toxic compound. The reduced graph representation of the OARG resulted in decreased performance in this application.

An advantage of the application of the kernel machine in this study is the low number of free parameters (between two and four, including the kernel).

[4] http://math.nist.gov/javanumerics/jama/

Therefore, we expect a suitable generalization of the models. A disadvantage is the lack of interpretability of models inferred by a Support Vector Machine.

We were able to improve the MSE as well as the Q^2 on the LOAEL benchmark problem with various setups. The number of free parameters is low for all approaches (two for ϵ-SVR and a maximum of two for the kernel function).

We presented a method for the prediction of chronic rat toxicity based on kernel machines. The benchmarks show that several of the tested methods with an error close to the estimated interlaboratory reproducibility of the animal experiments, which can be regarded as theoretical limit, are well-suited for predicting such data.

References

1. Azencott, C.-A., Ksikes, A., Joshua Swamidass, S., Chen, J.H., Ralaivola, L., Baldi, P.: One- to four-dimensional kernels for virtual screening and the prediction of physical, chemical, and biological properties. Journal of Chemical Information and Modeling 47(3), 965–974 (2007)
2. Chang, C.-C., Lin, C.-J.: Libsvm: A library for support vector machines (2001)
3. Cronin, M.T.D., Jaworska, J.S., Walker, J.D., Comber, M.H.I., Watts, C.D., Worth, A.P.: Use of qsars in international decision-making frameworks to predict health effects of chemical substances. Environmental Health Perspectives 10, 1391–1401 (2003)
4. Fröhlich, H., Wegner, J.K., Sieker, F., Zell, A.: Optimal assignment kernels for attributed molecular graphs. In: ICML, pp. 225–232 (2005)
5. Fröhlich, H., Wegner, J.K., Sieker, F., Zell, A.: Kernel functions for attributed molecular graphs - a new similarity-based approach to adme prediction in classification and regression. QSAR & Combinatorial Science 25, 317–326 (2006)
6. Gasteiger, J., Rudolph, C., Sadowski, J.: Automatic generation of 3d-atomic coordinates for organic molecules. Tetrahedron Computational Methods 3, 537–547 (1992)
7. Gower, J.C.: A general coefficient of similarity and some of its properties. Biometrics 27(4), 857–871 (1971)
8. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: ICML, pp. 321–328 (2003)
9. Leslie, C., Eskin, E., Noble, W.S.: The spectrum kernel: a string kernel for protein classification. In: Pacific symposium on biocomputing (2002)
10. Mahé, P., Ralaivola, L., Stoven, V., Vert, J.-P.: The pharmacophore kernel for virtual screening with support vector machines. Journal of Chemical Information and Modeling 46(5), 2003–2014 (2006)
11. Mazzatorta, P., Estevez, M.D., Coulet, M., Schilter, B.: Modeling oral rat chronic toxicity. Journal of Chemical Information and Modeling 48, 1949–1954 (2008)
12. Mumtaz, M.M., Knauf, L.A., Reisman, D.J., Peirano, W.B., DeRosa, C.T., Gombar, V.K., Enslein, K., Carter, J.R., Blake, B.W., Huque, K.I., Ramanujam, V.M.S.: Assessment of effect levels of chemicals from quantitative structure-activity relationship (qsar) models. i. chronic lowest-observed-adverse-effect level (loael). Toxicology Letters 79, 131–143 (1995)
13. Munro, I.C., Ford, R.A., Kennepohl, E., Sprenger, J.G.: Correlation of structural class with no-observed-effect levels: A proposal for establishing a threshold of concern. Food and Chemical Toxicology 34, 829–867 (1996)

14. Ralaivola, L., Swamidass, S.J., Saigo, H., Baldi, P.: Graph kernels for chemical informatics. Neural Networks 18(8), 1093–1110 (2005)
15. Rupp, M., Proschak, E., Schneider, G.: Kernel approach to molecular similarity based on iterative graph similarity. Journal of Chemical Information and Modeling 47(6), 2280–2286 (2007)
16. Saigo, H., Kadowaki, T., Tsuda, K.: A linear programming approach for molecular qsar analysis. In: International Workshop on Mining and Learning with Graphs (2006)
17. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)
18. Steinbeck, C., Han, Y., Kuhn, S., Horlacher, O., Luttmann, E., Willighagen, E.: The chemistry development kit (cdk): an open-source java library for chemo- and bioinformatics. Journal of Chemical Information and Computer Science 43(2), 493–500 (2003)
19. Talete srl, Milano, Italy. dragonX 1.4 for Linux (Molecular Descriptor Calculation Software)
20. Tilaoui, L., Schilter, B., Tran, L.-A., Mazzatorta, P., Grigorov, M.: Integrated computational methods for prediction of the lowest observable adverse effect level of food-borne molecules. QSAR & Combinatorial Science 26, 102–108 (2007)
21. Todeschini, R., Consonni, V., Mannhold, R., Kubinyi, H., Timmerman, H.: Handbook of Molecular Descriptors. Wiley-VCH, Weinheim (2000)
22. U. S. Environmental Protection Agency. ECOTOX User Guide: ECOTOXicology Database System. Version 4.0 (2006)
23. Venkatapathy, R., Moudga, C.J., Bruce, R.M.: Assessment of the oral rat chronic lowest observed adverse effect level model in topkat, a qsar software package for toxicity prediction. Journal of Chemical Information and Computer Sciences 44(5), 1623–1629 (2004)
24. Vert, J.-P.: The optimal assignment kernel is not positive definite (2008)
25. Walker, J.D., Carlsen, L., Hulzebos, E., Simon-Hettich, B.: Global government applications of analogues, sars and qsars to predict aquatic toxicity, chemical or physical properties, environmental fate parameters and health effects of organic chemicals. SAR and QSAR in environmental research 13, 607–616 (2002)

Simulating Evolution of *Drosophila Melanogaster Ebony* Mutants Using a Genetic Algorithm

Glennie Helles

University of Copenhagen,
Universitetsparken 1, 2100 Copenhagen, Denmark
glennie@diku.dk
http://www.diku.dk/~glennie

Abstract. Genetic algorithms are generally quite easy to understand and work with, and they are a popular choice in many cases. One area in which genetic algorithms are widely and successfully used is artificial life where they are used to simulate evolution of artificial creatures. However, despite their suggestive name, simplicity and popularity in artificial life, they do not seem to have gained a footing within the field of population genetics to simulate evolution of real organisms – possibly because genetic algorithms are based on a rather crude simplification of the evolutionary mechanisms known today. However, in this paper we report how a standard genetic algorithm is used to successfully simulate evolution of *ebony* mutants in a population of *Drosophila melanogaster* (D.*melanogaster*). The results show a remarkable resemblance to the evolution observed in real biological experiments with *ebony* mutants, indicating that despite the simplifications, even a simple standard genetic algorithm does indeed capture the governing principles in evolution, and could be used beneficially in population genetics studies.

Keywords: Genetic algorithms, simulation, population genetics.

1 Introduction

Darwin's book "On the Origin of the Species" was published in 1859 and even though evolution was already generally a recognized phenomenon amongst scientists at the time, Darwin offered a new way to explain evolution that has since been broadly accepted by the scientific community.

Genetic algorithms can generally be thought of as a formalization of Darwin's evolutionary theory. Originally formulated by John Holland in the 1960's, genetic algorithms are a set of heuristics that are perhaps mostly known for their ability to find optimal or near-optimal solutions in large (possibly infinitely large) search spaces [4,7] and their use in artificial life [5]. They come in many different shapes and forms, but the underlying framework involving an encoding-, selection- and breeding-strategy is the same for all of them.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 37–43, 2009.

In this experiment we use a genetic algorithm to simulate a population of *Drosophila melanogaster* (D.*melanogaster*), also known as the common fruit fly. The fly is with its approximately 13.600 genes a relatively complex animal with a fairly short life cycle of roughly 30-40 days [3], although flies kept in laboratories usually have a slightly shorter life span[3]. This makes the fruit fly ideal for studies of population genetics and it is indeed widely used for just that [10].

One of the genes that has been throughly studied is the *ebony* gene [10], which aside from being associated with the color of the fly, influences both vision and movement and has a direct influence on the breeding success of the fly. Furthermore, only approximately 80% of *ebony* mutants ever hatch giving the mutants a natural disadvantage [10].

Each fly has two instances of each chromosome, which gives rise to three distinct *ebony* genotypes: the mutant genotype (e/e) has a mutation in the *ebony* gene on both chromosomes, the heterozygous genotype (+/e) has a mutation in the *ebony* gene on only one of the chromosomes while the wild type genotype (+/+) has no mutation in the *ebony* gene. The *ebony* gene is a recessive gene, which generally means that only flies with the mutant genotype will show the mutant characteristics. In practice, however, many of the heterozygous flies can easily be distinguished upon visual inspection and mating studies also shows that the flies are definitely able to distinguish between all three genotypes[8]. From a population geneticist's point of view, the *ebony* gene is interesting because hatching along with the ability to successfully move around and breed are generally considered extremely important for a fly, if the genes are to be preserved in future generations. Aside from their lower hatching success, the ability of the *ebony* mutant to move around is also impaired compared to both the wild type fly and especially the heterozygous fly, just as the mutants eyesight is quite limited [10]. A male fly's ability to move around quickly is thought to be an attractive characteristic among female flies, and good vision will naturally allow the flies to spot the most attractive mating partner from a longer distance.

However, the male mutant fly has a larger mating capacity (it can inseminate more females in a 24 hour period) than the wild type counter part [8,9] and the female mutant flies tend to be less selective when being courted by a male fly – possibly because of their impaired vision. Heterozygous male flies have the largest mating capacity of all the genotypes, which would seem to give this genotype an advantage, but the heterozygous female flies on the other hand accepts the fewest mating invitations, thereby apparently canceling out the advantage. Biological experiments, wherein the initial population contains 25%, 50% and 75% mutants respectively, have shown that the frequency of mutants always drop rapidly during the first 5-10 generations, but then stabilizes (see [8,9] and Figure 1 A).

A number of systems for simulating and predicting allele frequencies of genes exists. Very simple systems based on Wright-Fisher populations assume neutral development and are thus too imprecise for studying real systems. Other non-neutral systems that use Markov chains have been applied [2] but to our knowledge genetic algorithms have for some reason, despite their otherwise suggestive name, not previously been used in population genetics simulations. The appeal

of using a genetic algorithm for a population geneticist is that the terminology is highly familiar and thus very intuitive. We realize that genetic algorithms have been developed and extended much over the years as our understanding of evolution has increased [1], but this type of experiment relates to the very fundamentals of evolution and we thus believe that, aside from being very simple, the original genetic algorithm is quite adequate.

2 Methods

The genetic algorithm used for the simulation is constructed such that each individual (fly), referred to as a D.*simulation*, has two chromosome pairs explicitly modeled. The chromosome 3 pair, which includes the *ebony* gene, and the sex chromosome pair (X/Y), which determines the sex of the fly. All flies possess the following basic characteristics: they can fly, walk, see and breed. Furthermore, we have introduced a "resting period" for the male D.*simulation*, which is triggered by the mating procedure, to ensure that the simulated fly has the same mating capacity as the real fly. Female flies do not have a resting period, but they do have the ability to lay eggs. How far they can fly, walk or see depends on the genotype of the fly (see Table 1) just as it is the case for the real fly and likewise for the required resting period of the male flies and the selectivity of the female flies. During the resting period the male fly is unable to mate, but it may still move around. How many cells the simulated flies can move and see is based on our interpretation of how the different genotype compared to each other.

D.*melanogaster* goes through several stages before it becomes a real fly [3]. However, the egg stage, larval stage and pupa stage are for the most part uninteresting when looking at population genetics and D.*simulation* thus only has one stage (the egg stage) before it becomes a fly. Development from egg to fly for D.*melanogaster* take roughly 14 days and the length of the egg stage for D.*simulation* is thus chosen randomly between 12 and 16 days. The egg never moves. Like the D.*melanogaster ebony* mutant, a D.*simulation ebony* mutant egg has only a 80% chance of ever hatching. Eggs that do not hatch are removed from the world. After hatching the D.*simulation* flies are sexually active after 12 hours and the female flies start laying eggs after 2 days. These values are equal to

Table 1. For each genotype (+/+ = wild type, +/e = heterozygous and e/e = mutant) is indicated the maximum flying and walking distance, how far they can see, the mating capacity of the males and the courtship acceptance frequency of the females

	+/+	+/e	e/e
Walk (cells)	5	5	4
Fly (cells)	12	12	8
Sight (cells)	5	5	3
Mating capacity (24 hours)	3	6	4
Courtship acceptance (%)	60	50	65

those observed in real experiments [3]. Real female flies may lay several hundreds of eggs – whether they are fertilized or not – over a period of approximately 10 days, but in order to control population size a female D.*simulation* lays only between 1 and 6 eggs. All unfertilized eggs are removed immediately from the environment. The complete life cycle of D.*simulation* including the egg stage is chosen randomly between 28 and 32 days. The initial population contains 100 flies and 20 eggs (50% of each sex). If the population grows to include more than 250 flies, a 1% chance of "sudden death" is introduced.

The simulation uses a 3-dimensional world containing 50 x 50 x 50 cells. Each cell can only hold one fly or egg. Each time step is equivalent to one hour and the flies can perform one movement per time step. How far a fly moves in each time step is chosen randomly, although it never exceeds the distance stated in Table 1. If a cell is already occupied by another fly a new movement and distance is chosen. Mating occurs at the end of each time step. Female flies are selected with a frequency that match their "courtship acceptance" and for each selected female fly a male fly is chosen by using a fitness-proportionate selection strategy known as a "roulette wheel" on the non-resting male flies that occupy cells within eyesight distance of the female. The Roulette Wheel ensures that male flies with higher fitness values are chosen more frequently than male flies with lower fitness values.

The breeding strategy uses only mutation (not crossover), as the simulation is only focused on the *ebony* gene. Spontaneous mutation of the *ebony* gene occurs with a frequency of $8x10^{-4}$. The mutation frequency is set 40 times higher than what is typically observed in real life to compensate for the inclusion of only one type of *ebony* mutants, as opposed to the 40 different mutations to the *ebony* gene that has been identified in the real fly.

3 Results

The fitness values upon which these results are based are listed in Table 2. They are the result of simply adding the values from Table 1, except "courtship acceptance", which is purely a female trait that does not influence the male fitness value. The simulation is run 10 times of 50000 time steps ($\approx$ 5.7 years) for each start population of 25%, 50% and 75% mutants respectively. Due to the stochastic aspect of genetic algorithms, different results may be obtained for each run and running the algorithm multiple times was thus done.

Table 2. Each of the fly's characteristics contributes to the overall fitness value, but varies for every genotype (+/+ = wild type, +/e = heterozygous and e/e = mutant)

	Fitness
+/+	25
+/e	28
e/e	19

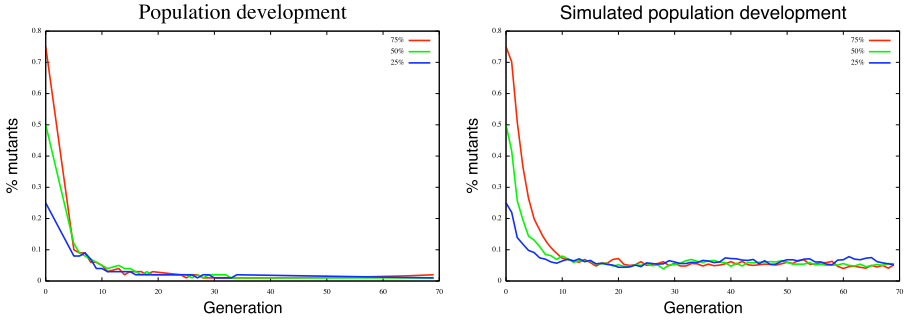


Fig. 1. The results taken from the real experiments compared to the results from the simulation. The simulated result shown for each start population of 25%, 50% and 75% mutants is the average of the 10 runs.

Both the results from real experiments with D.*melanogaster* and the results from the simulation are presented in Figure 1. The results from the real experiment are taken directly from [10]. The figure shows that the frequency of mutants in both the D.*melanogaster* and D.*simulation* population drop very rapidly initially and then stabilizes after roughly 10-15 generations. As can be seen, development of the gene in the simulated population matches the real experiment close to perfectly.

It should of course be emphasized that selection of flies are based on the very simple rules that are defined in section 2 and they remain the same throughout the simulation. The composition of genotypes in the population at a given time is not considered and does not in any way affect which flies are selected for breeding. In other words, we do not have any special rules that are applied if mutants appear to either take over the population or die out.

4 Discussion

The main focus of this experiment is to determine if a standard genetic algorithm is able to successfully capture the fundamental aspects of evolution as can be observed in population genetics experiments. By inspecting the mutant frequency from experimental results with D.*melanogaster* (Figure 1 A), two things stand out: the initial drastic drop in mutant frequency and the following stabilization. Both traits are clearly observed in the simulated population (Figure 1 B). One could fear that the mutants would either completely take over the population or alternatively simply die out, as indeed seems very likely when observing the initial drastic drop, but the pattern resembles the pattern observed in nature. With the settings given in Table 1 and 2 the genetic algorithm does thus appear to very successfully capture the governing aspect of evolution and would in fact seem to be an obvious choice for simulating population genetics.

Another interesting observation is that, in Søndergaard [9] they describe how females tend to pick differently depending on the male compositions in the population, but we do not actually have to explicitly consider that in the simulation. The females follow that same rules at all times during the simulation regardless of the male composition in the population, and the behavior thus seem to emerge naturally simply from each individual following a few simple rules. This phenomenon is also known in swarm intelligence where simulation have clearly demonstrated that the seemingly intelligent movement of flocking animals in fact emerges from the simple set of rules that each animal follow [6].

When genetic algorithms are as accurate, as seen here, they can be used not only to simulate evolution but also to investigate underlying biological aspects. It is for example quite interesting to note that although the *ebony* gene is recessive, and one would thus expect that the heterozygous fly and the wild type fly should be viewed as equal, it proved to be imperative for the accuracy of the simulation to distinguish between the two genotypes. If such a distinction was not made the drop in mutant frequency was generally more linear, and most often the mutants died out before 50000 time steps.

We wish to emphasize that we do not suggest that simulation with a standard genetic algorithm can be used to simulate evolution of, say, the effect of a new mutation - the genetic algorithm relies heavily on prior knowledge that is relevant for calculating fitness of the fly. However, the effect of changing for instance the start compositions of flies (such as more mutants or fewer males) or even introducing impairments (such as restrictions on walking distance) could easily be studied using a simulation as this. Also, an analysis of exactly how much the fitness values can be varied while maintaining the characteristic development pattern could be done to help us understand how robust nature really is and indicate how big an advantage or disadvantage each genotype can have without altering the population development pattern. Running the computer simulation for 50.000 time steps (corresponding to $\approx$5.7 years or 70 generations) with our parameter settings takes just under 5 minutes on a 2.2GHz Intel processor.

As a final note, it should be mentioned, that this simulation is rather simple, as it involves only mutation as a biological operator. The other major biological operator, crossover, is not really used in this experiment, and we are thus not able to draw any conclusions about how well genetic algorithms would for instance simulate evolution of couple genes, but given the good results we have achieved here with relatively little effort, we are quite optimistic, and it would be interesting to attempt a more complicated simulation.

5 Conclusion

The results from experiments with D.*melanogaster* and the results from the simulated fly, D.*simulation*, bear a remarkable resemblance. Despite being based on a simplification of evolution as we know it, genetic algorithms do appear to be quite able to capture the fundamental aspects of evolution. The prospect of using genetic algorithms in population genetics where some knowledge about

fitness have already been established thus look very good and further investigation into more complicated simulations of, for instance, coupled genes in the D.*melanogaster*, would be interesting to carry out.

References

1. Banzhaf, W., Beslon, G., Christensen, S., Foster, J.A., Képès, F., Lefort, V., Miller, J.F., Radman, M., Ramsden, J.J.: Guidelines: From artificial evolution to computational evolution: a research agenda. Nature Reviews Genetics 7, 729–735 (2006)
2. Fearnhead, P.: Perfect Simulation From Nonneutral Population Genetic Models: Variable Population Size and Population Subdivision. Genetics 174, 1397–1406 (2006)
3. Griffiths: An Introduction to Genetic Analysis. Freeman, New York (1996)
4. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1999)
5. Mitchell, M., Forrest, S.: Genetic algorithms and artificial life. Artificial Life 1, 267–289 (1994)
6. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. Computer Graphics 21, 25–34 (1987)
7. Russel, Norvig: Artificial Intelligence - a modern approach. Prentice Hall, Englewood Cliffs (1995)
8. Søndergaard, L.: Mating competition on artificial populations of Drosophila melanogaster polymorphic for ebony. Heraditas 103, 47–55 (1985)
9. Søndergaard, L.: Mating competition on artificial populations of Drosophila melanogaster polymorphic for ebony II. Heraditas 103, 47–55 (1985)
10. Søndergaard, L., Sick, K.: Long-term frequencies of *ebony* in artificial Drosophila melanogaster populations kept in light and darkness. Heraditas 103, 57–61 (1985)

Microarray Biclustering: A Novel Memetic Approach Based on the PISA Platform

Cristian Andrés Gallo[1], Jessica Andrea Carballido[1], and Ignacio Ponzoni[1,2]

[1] Laboratorio de Investigación y Desarrollo en Computación Científica (LIDeCC),
Departamento de Ciencias e Ingeniería de la Computación,
Universidad Nacional del Sur, Av. Alem 1253, 8000, Bahía Blanca, Argentina
{cag,jac,ip}@cs.uns.edu.ar
[2] Planta Piloto de Ingeniería Química (PLAPIQUI) - UNS – CONICET
Complejo CRIBABB, Co. La Carrindanga km.7, CC 717, Bahía Blanca, Argentina

Abstract. In this paper, a new memetic approach that integrates a Multi-Objective Evolutionary Algorithm (MOEA) with local search for microarray biclustering is presented. The original features of this proposal are the consideration of opposite regulation and incorporation of a mechanism for tuning the balance between the size and row variance of the biclusters. The approach was developed according to the Platform and Programming Language Independent Interface for Search Algorithms (PISA) framework, thus achieving the possibility of testing and comparing several different memetic MOEAs. The performance of the MOEA strategy based on the SPEA2 performed better, and its resulting biclusters were compared with those obtained by a multi-objective approach recently published. The benchmarks were two datasets corresponding to *Saccharomyces cerevisiae* and *human B-cells Lymphoma*. Our proposal achieves a better proportion of coverage of the gene expression data matrix, and it also obtains biclusters with new features that the former existing evolutionary strategies can not detect.

Keywords: Gene regulation, biclustering, evolutionary algorithms, PISA.

1 Introduction

The study of complex interactions between macro-molecules during transcription and translation processes constitutes a challenging research field, since it has a great impact in various critical areas. In this context, the microarray technology arose as a fundamental tool to provide information about the behavior of thousands of genes. The information provided by this technology corresponds to the relative abundance of the mRNA of genes under a given experimental condition. The abundance of the mRNA is a metric that can be associated to the expression level of the gene. This information can be arranged into a matrix, namely gene expression data matrix, where rows and columns correspond to genes and experiments respectively. Each matrix entry is a real number that represents the expression level of a given gene under a given condition.

An important issue in gene expression data analysis consists in grouping genes that present a similar, or related, behavior according to their expression levels. The

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 44–55, 2009.

achievement of this task helps in inferring the functional role of genes during protein transcription. Therefore, based on the data about the relations between genes and their products, the gene regulatory networks (GRNs) can be discovered.

In general, during the process of identifying gene clusters, all of the genes are not relevant for all the experimental conditions, but groups of genes are often co-regulated and co-expressed only under some specific conditions. This important observation has leaded the attention to the design of biclustering methods that simultaneously group genes and samples [1]. In this context, a satisfactory bicluster consists in a group of rows and columns of the gene expression data matrix that satisfies some similarity score [2] in conjunction with other criteria.

In this paper, we propose a memetic multi-objective evolutionary approach implemented in the context of the PISA platform to solve the problem of microarray biclustering. Our technique hybridizes traditional multi-objective evolutionary algorithms (MOEAs) with a new version of a well-known Local Search (LS) procedure. To the best of our knowledge, this methodology introduces two novel features that were never addressed, or partially dealt-with, by other evolutionary techniques designed for this problem instance. The first contribution consists in the design of the individual representation that contemplates the mechanisms of opposite regulation. The other new characteristic is the incorporation of a mechanism that controls the trade-off between size and row variance of the biclusters. The rest of the paper is organized as follows: in the next section some concepts about microarray biclustering are defined; then, a brief review on existing evolutionary methods used to tackle this problem is presented; in Section 4 our proposal is introduced; then, in Section 5, all the experimental framework and the results are put forward; finally some conclusions are discussed.

2 Microarray Biclustering

As it was aforementioned, expression data can be viewed as a matrix **E** that contains expression values, where rows correspond to genes and columns to the samples or *conditions*, taken at different experiments. A matrix element e_{ij} contains the measured expression value for the corresponding gene i and sample j. In this context, a bicluster is defined as a pair (G, C) where $G \subseteq \{1, \ldots, m\}$ is a subset of genes (rows) and $C \subseteq \{1, \ldots, n\}$ is a subset of conditions (columns) [2]. In general, the main goal is to find the largest bicluster that does not exceed certain homogeneity constrain. It is also important to consider that the variance of each row in the bicluster should be relatively high, in order to capture genes exhibiting fluctuating coherent trends under some set of conditions. The bicluster size is the number of rows $f(G)$ and the number of columns $g(C)$. The homogeneity $h(G,C)$ is given by the mean squared residue score, while the variance $k(G,C)$ is the row variance [2]. Therefore, our optimization problem can be defined as follows:

maximize

$$f(G) = |G| \cdot \tag{1}$$

$$g(C) = |C| \cdot \tag{2}$$

$$k(G,C) = \frac{\sum_{g \in G, c \in C} \left(e_{gc} - e_{gC}\right)^2}{|G| \cdot |C|} \cdot \tag{3}$$

subject to

$$h(G,C) \leq \delta \cdot \tag{4}$$

with $(G,C) \in X$, $X = 2^{\{1,\ldots,m\}} \times 2^{\{1,\ldots,n\}}$ being the set of all biclusters, where

$$h(G,C) = \frac{1}{|G| \cdot |C|} \sum_{g \in G, c \in C} \left(e_{gc} - e_{gC} - e_{Gc} + e_{GC}\right)^2 \cdot \tag{5}$$

is the mean squared residue score,

$$e_{gC} = \frac{1}{|C|} \sum_{c \in C} e_{gc}, \quad e_{Gc} = \frac{1}{|G|} \sum_{g \in G} e_{gc} \cdot \tag{6,7}$$

are the mean column and row expression values of (G,C) and

$$e_{GC} = \frac{1}{|G| \cdot |C|} \sum_{g \in G, c \in C} e_{gc} \cdot \tag{8}$$

is the mean expression value over all the cells that are contained in the bicluster (G,C). The user-defined threshold δ represents the maximum allowable dissimilarity within the cells of a bicluster. In other words, the residue quantifies the difference between the actual value of an element e_{gc} and its expected value as predicted for the corresponding row mean, column mean, and bicluster mean. If a bicluster has a mean square residue lower than a given value δ, then we call the bicluster a δ-bicluster. The problem of finding the largest square δ-bicluster is NP-hard [2]. The high complexity of this problem has motivated researchers to apply various approximation techniques to generate near optimal solutions. In particular, evolutionary algorithms (EAs) are well-suited for addressing this class of problems [3, 4, 5].

3 Microarray Biclustering with Evolutionary Algorithms

The first reported approach that tackled microarray biclustering by means of an EA was proposed by Bleuler *et al.* [5]. In this work, several variants are presented. They analyze the use of a single-objective EA, an EA combined with a LS strategy [2] and the LS strategy alone [2]. In the case of the EA, one novelty consists in a form of diversity maintenance that can be applied during the selection procedure. For the case of the EA hybridized with a LS strategy, they consider whether the new individual yielded by the LS procedure should replace the original individual (*Lamarckian* approach) or not (*Baldwinian* approach). As regards the LS as a stand alone strategy, they propose a new non-deterministic version, where the decision on the course of execution is made according to some probability.

Regarding the EA, a binary representation for the individuals where each individual stands for a given bicluster is adopted, and independent bit mutation and uniform crossover are used. For the definition of the fitness function, they distinguish two cases: whether the EA operates alone or if it works together with the LS strategy. For the first

situation a better fitness value, obtained from the size of the bicluster, is assigned to those individuals that comply with the residue restriction. If the bicluster has a residue over a given threshold, namely δ, then a value greater than 1 is set. For the second case, as the residue constraint is considered by the LS strategy, they only look at the size of the biclusters for the fitness assignment. For the experiments, two datasets were used: *Yeast* [6] and *Arabidopsis thaliana* [7, 8]. The study of the results is organized considering whether the aim is to get a unique bicluster or a set of biclusters. For the analysis of a single bicluster, the evaluation is focused on the size of the biclusters, and the algorithm that performed better was the EA combined with the LS method by means of an updating policy. For the second case of analysis, a comparison of the results as regards the covering of matrix **E** is performed, and the hybridized EA with diversity maintenance combined with LS did better in this sense.

Another approach, called SEBI for Sequential Evolutionary BIclustering, was later proposed by Divina and Aguilar-Ruiz [4]. In this work, an EA is presented where the individuals represent biclusters by means of binary strings. The main idea of this sequential technique is that the EA is run several times. From each run, the EA yields the best bicluster according to its size, row variance and overlapping factors. If its residue value (as defined by Chung and Church [2]) is lower than δ, then the bicluster is added into an archive that they call *Results*. Whenever this is the case, the method keeps track of the elements of the bicluster so as to use this information to minimize overlapping during the next run of the EA.

As regards the details of the EA, the fitness function combines the aforementioned objectives by means of a non-Pareto aggregative function. Tournament selection is chosen and several options for the recombination operators were implemented. For the experimental studies, the EA was executed for two datasets: *Yeast* [6] and *Human B-cells* [9]. The comparison is performed against the biclusters found by Chung and Church as regards the covering of the whole gene expression matrix **E**. For the *Yeast* dataset, SEBI manages to cover 38% of **E**, while Chung and Church's covers 81%. Regarding the *Human* dataset, SEBI covers 34% while Chung and Church's biclusters cover 37%. The authors consider that these results can be explained as a consequence of the overlapping factor, since the consideration of this objective naturally goes in detriment of the other goals.

Finally, Mitra and Banka [3] present a MOEA combined with a LS [2] strategy. This method constitutes the first approach that implements a MOEA based on Pareto dominancy for this problem. The authors base their work on the NSGA-II, and look for biclusters with maximum size and homogeneity. The individual representation is the same as in the previously introduced methods; and uniform single-point crossover, single-bit mutation and crowded tournament selection are implemented. The LS strategy is applied to all of the individuals with a *Lamarkian* approach, at the beginning of every generational loop. The method is tested on microarray data consisting of two benchmark gene expression datasets, *Yeast* and *Human B-cell Lymphoma*. For the analysis of the results, a new measure called Coherence Index (CI) is introduced. CI is defined as "the ratio of mean squared residue score to the size of the formed bicluster". The biclusters are compared to those reported by Chung and Church and, in all the cases, Mitra and Banka's results indicate a better performance in terms of the bicluster size, while satisfying the homogeneity criterion in terms of δ. However, as regards coverage, Chung and Church's work produces better results.

4 Our Proposal

The aim of our study is to use a MOEA for approximating the Pareto front of biclusters from a given gene expression matrix, as this approach gives the best tradeoff between the objectives that we want to optimize. However, in view of the fact that the Pareto front also includes biclusters that do not satisfy the homogeneity restriction, we need to guide the search to the area where this restriction is accomplished. In that context, we apply a LS technique based on Chung and Church's procedure after each generation, thus orienting the exploration and speeding up the convergence of the MOEA by refining the chromosomes. Besides, the results achieved by other authors [3, 5] reveal that MOEAs alone obtain poor biclusters.

In order to consider inverted rows, we have extended the classical representation of a bicluster and we have also modified the genetic operators. Then, our proposal performs over a double-sized search space, in contrast with the evolutionary biclustering methods found in the literature [3, 4, 5]. The importance of including these inverted rows resides in that they form *mirror images* of the rest of the rows in the bicluster, and can be interpreted as opposite co-regulated [2]. In this way, our proposal is able to find biclusters that the former evolutionary methods cannot detect.

As regard to the implementation, the multi-objective strategy was built on the base of a platform called PISA [10]. PISA is a text-based interface for search algorithms. It splits an optimization process into two modules. One module contains all the parts that are specific to the optimization problem (e.g., evaluation of solutions, problem representation, and variation of solutions) and is called the *Variator*. The other module contains the parts of an optimization process which are independent of the optimization problem (mainly the selection process). This part is called the *Selector*. These two modules are implemented as separate programs which communicate through text files.

For this work, we have designed a *Variator* specific for the microarray biclustering application, and we have combined it with the *Selectors* corresponding to the IBEA [11], NSGAII [12] and SPEA2 [13] optimization algorithms. The reason for the selection of these MOEAs is that they are the most recommended evolutionary optimizers in the literature. In this way, we will assess the MOEA that exhibits the best performance for the problem. In the following sections, we will describe the main features of the implemented *Variator* and how the LS is incorporated into the search process.

Individual's Representation

Each individual represents one bicluster, which is encoded by a fixed size string built by appending a string for genes with another bit string for conditions. The individual corresponds to a solution for the problem of optimal bicluster generation. If a string position (*locus*) is set to 1, it means that the relative row or column belongs to the encoded bicluster, otherwise it does not. To take into account the inverted rows we also considerer the addition of negative values in the string for genes. That is to say, a *locus* of the string is set to -1 when the relative inverted row belongs to the encoded solution. Figure 1 shows an example of such encoding for a random individual.

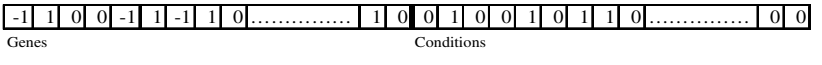


Fig. 1. An encoded individual representing a bicluster

Genetic Operators

It is important to give a brief description of the genetic operators, since they have a key influence in how the search is performed by the MOEA.

Mutation. This operator is implemented in the following way: first it is determined if the individual needs to be mutated by means of the probability assigned to the operator. In such case, a position of the string is selected at random, then proceeding to alter the *locus* in question. If the resulting position is a column, the corresponding *locus* is simply complemented. On the other hand if the resulting position is a row, then we have two cases: if the *locus* is set to 0 then it is set to 1, and the sign is determined with a probability of 0.5. If the *locus* is in on (1 or -1) we simply change the value to 0.

Recombination. A two-point crossover was implemented with a little restriction: one random point is selected on the rows and the other random point is selected on the columns. In this way, we ensure that the recombination is performed over both the genes and the conditions subspaces. Then, when both children are obtained combining each one of the two parents' parts (i.e. the ends and the center), the individual that is selected to be the only descendant is the non-dominated one. If both are non-dominated, one of them is chosen at random.

Multi-Objective Fitness Function

As regards the objectives to be optimized, we observed that it was necessary to generate maximal sets of genes and conditions, while maintaining the "homogeneity" of the bicluster with a relatively high row variance. These bicluster features, conflicting to each other, are well-suited for multi-objective modeling. In that context, we decided to optimize the objectives defined by equations 1- 4 (see *Section 3*): the quantity of genes, the quantity of conditions, the row variance, and the mean squared residue. The first three objectives are maximized, while the last one is minimized.

Local Search

This subsection describes the LS procedure that hybridizes the selected MOEAs. The LS is applied into the *Variator* side to the biclusters that are selected by the *Selector* as the resulting individuals of each generation. Adding the LS to the *Variator* is the only way to hybridize a MOEA without altering the basic principles of PISA [10]. The greedy approach is based on Chung and Church's work [2], with some modifications introduced in order to consider the row variance and the overall efficiency of the proposal. The algorithm starts from a given bicluster (G,C). The genes or conditions having mean squared residue above (or below) a certain threshold are selectively eliminated (or added) according to Algorithm 1.

The main differences with Chung and Church's implementation are the following:

- In Step 3, we remove multiple nodes considering a different threshold, $\alpha.\delta$ instead of $\alpha.h(G,C)$. As a consequence, Step 5 is performed a smaller number of times with respect to the original proposal. This is useful because, with a proper setting of the parameter α, the CPU time needed to optimize a bicluster is decreased. This is possible without loosing significant precision of the algorithm.
- In Step 9, we incorporated the row variance, adding the rows that will increase in a certain proportion the overall row variance of the individual.

- Finally, in the Steps 7-9, the original algorithm tries to add each row, each column and each inverted row, in that order. In our case, we first attempt to add each condition. This increases (on average) the amount of conditions of the resulting bicluster since a column, in general, has more probability of being inserted in the solution if it contains less quantity of rows.

Beside δ, two additional parameters need to be set for this algorithm. α determines how often multiple gene deletion is used. A higher α leads to less multiple gene deletion and thus, in general, requires more CPU time. The other parameter is μ that establishes a relationship between the number of genes and the row variance of the bicluster. A bigger μ results in individuals with a higher row variance and a smaller size. If $\mu = 0$, this step results equivalent to that of the original proposal.

Algorithm 1. Local Search

Input: (G, C) *(a bicluster)*

Output: (G, C)' *(an improved bicluster)*

Step 1: *Compute* e_{gC}, e_{Gc}, e_{GC} *and* $h(G, C)$ *by equations 5-8.*

Step 2: *if* $h(G, C) < \delta$ *go to step 7.*

Step 3: *Remove all genes* $i \in G$ *satisfying* $\frac{1}{C}\sum_{c \in C}\left(e_{gc} - e_{gC} - e_{Gc} + e_{GC}\right)^2 > \alpha \cdot \delta$

Recalculate all means and perform the same operation on conditions. The equation for conditions is analogous.

Step 4: *Recompute* e_{gC}, e_{Gc}, e_{GC} *and* $h(G, C)$. *If* $h(G, C) < \delta$ *go to step 6.*

Step 5: *Remove node* i *with the largest* $d(i) = \frac{1}{C}\sum_{c \in C}\left(e_{ic} - e_{iC} - e_{Gc} + e_{GC}\right)^2$

The equation for the conditions is analogous. Go to step 4.

Step 6: *Recompute* e_{gC}, e_{Gc}, e_{GC} *and* $h(G, C)$.

Step 7: *Add all conditions* $c \notin C$ *satisfying* $\frac{1}{G}\sum_{g \in G}\left(e_{gc} - e_{gC} - e_{Gc} + e_{GC}\right)^2 \leq h(G,C)$

Step 8: *Recompute* e_{gC}, e_{Gc}, e_{GC} *and* $h(G, C)$ *and* $k(G, C)$, *this last by means of equation 3.*

Step 9: *Add all genes* $g \notin G$ *(or its inverse) satisfying*

$$\frac{1}{C}\sum_{c \in C}\left(e_{gc} - e_{gC} - e_{Gc} + e_{GC}\right)^2 \leq h(G,C) \;\wedge\; k(G \cup \{g\}, C) \geq \mu \cdot k(G,C)$$

The equation for the inverse only differs in that the term e_{gc} *is multiplied by -1.*

5 Experimental Framework and Results

Two different goals were established for our study. First we need to determine which of the memetic MOEAs performs better in this class of problem. The analysis will be performed with the tools provided in Knowles *et al.* [14]. Then, we will compare the selected memetic evolutionary algorithm with the approach of Mitra and Banka [3] since, to the best of our knowledge, this is the only multi-objective evolutionary method for microarray biclustering found in the literature.

Performance Assessment

As it was aforementioned, the choice of the best memetic MOEA will be based on the results of the tools provided in Knowles *et al.* [14], which are well recognized in the

area of multi-objective optimization. The metrics applied in the evaluation of the MOEAs are the *Dominance Ranking* [14], the *Hypervolumen Indicator* I_H^- [15], the multiplicative version of the *Unary Epsilon Indicator* I_ε^1 [16], and the *R2 Indicator* I_{R2}^1 [17]. The *Dominance Ranking* is useful in the assessment of quite general statements about the relative performance of the considered optimizers, since it merely relies on the concept of Pareto dominance and some ranking procedure. The *Quality Indicator I* measures the number of goals that have been attained for the optimizers being under consideration. Each one of the indicators empathize a different aspect in the preference of the solutions obtained. For the details of the previous metrics please be referred to Knowles *et al.* [14].

If a significant difference can be demonstrated using the *Dominance Rank*, the only purpose of the *Quality Indicators* is to characterize further differences in the approximation sets. On the other hand, if we cannot establish significant differences by means of the *Dominance Rank*, then the *Quality Indicators* can help us in the decision of which one of the optimizers is better. However, these results do not confirm that the selected method generates the better approximation sets. In order to make inferences about the results of the previous metrics we will apply the *kruskal-wallis test* [18], since more than two methods are tested [14].

For this analysis, we have used two microarray datasets, the *Saccharomyces cerevisiae* cell cycle expression data from [6] and the *human B-cells Lymphoma* expression data from [9]. The yeast data contain 2.884 genes and 17 conditions, and the expression values denote relative mRNA abundance. All values are integers in the range between 0 and 600 replacing the missing values by 0. The *Lymphoma* dataset contains 4.026 genes and 96 conditions. The expression levels are integers in the range between -750 and 650, where the missing values were also replaced by 0. These datasets have been directly used as in [2].

First Experimental Phase

The three memetic MOEAs, IBEA, NSGA-II and SPEA2, have been evaluated with 50 runs and 75 generations over the two datasets. The Table 1 summarizes the parameters used in this benchmark. These values were selected from a few preliminary runs. In the case of the LS setup, δ was set with the same value as in [2], μ was set for the best tradeoff between size and row variance, and α was set considering the overall efficiency on each dataset. All the executions were controlled by the *Monitor* module [10]. In the case of the IBEA algorithm, we chose the *Additive Epsilon Indicator*, and the rest of the parameters were set to the default values. Since PISA assumes that all the objectives are minimized, the four objectives of our approach (see *equations 1-4*) were adapted accordingly. For the parameters of the indicators, we maintained the default values of the nadir and ideal points (appropriately extended to four objectives), since the objectives are automatically normalized to the interval [1..2] by the tools.

Table 1. Parameter's settings for this study

		Generations	Mutation Prob.	Crossover Prob.	δ	α	μ
Our variator	*Yeast*	75	0,3	0,9	300	1,8	0,998
	Lymphoma				1200	1,5	0,999

PISA	α	μ	λ
	100	50	50

Table 2. *Kruskal-wallis test* over the *Quality Indicators* I_H^- (left), I_ε^1 (middle) and I_{R2}^1 (right)

Hypervolumen Indicator

	IBEA	SPEA2	NSGA-II
IBEA	-	1	0,99
SPEA2	1,50E-07	-	0,16
NSGA2	1,09E-05	0,84	-

Multiplicative Epsilon Indicator

	IBEA	SPEA2	NSGA-II
IBEA	-	0,99	0,99
SPEA2	2,10E-04	-	0,61
NSGA2	2,10E-04	0,39	-

R2 Indicator

	IBEA	SPEA2	NSGA-II
IBEA	-	1	0,99
SPEA2	8,00E-09	-	0,09
NSGA2	4,00E-06	0,91	-

Table 3. Average of the objective's values of IBEA, SPEA2 and NSGA-II on the *Yeast* dataset (above) and on the *Lymphoma* dataset (below).

Yeast dataset

	average rows	average columns	average residue	average variance	average size
IBEA	1047,63	12,52	261,61	296,35	13116,33
SPEA2	794,59	10,37	224,31	296,47	8239,898
NSGA-II	646,34	9,92	204,75	236,04	6411,693

Lymphoma dataset

	average rows	average columns	average residue	average variance	average size
IBEA	655,93	60,71	1089,61	1135,93	39821,51
SPEA2	727,74	52,63	1048,91	1112,03	38300,96
NSGA-II	583,8	54,34	1046,68	1061,7	31723,69

As regards the experimental results, the *kruskal-wallis test* can not detect significant differences on the *Dominace Ranking* of the three MOEAs, assuming a statistically significant level $\alpha = 0.05$. This situation is equal for both datasets. In fact, all the results of the executions are assigned to the higher rank, showing that none of the MOEAs generates better approximation sets with respect to the others. This demonstrates the high influence in the search process of the LS and how it guides the MOEAs to the same areas on the search space. Table 2 shows, for the *Yeast* dataset, the results of the *kruskal-wallis test* over three *Quality Indicators*. The table contains, for each pair of optimizers O_R (row) and O_C (column), the p-values with respect to the alternative hypothesis that the *Quality Indicator I* is significantly better for O_R than for O_C. For the *Lymphoma* dataset, differences between the algorithms are discovered, but none of them are statistically significant ($\alpha = 0.05$).

As it is shown in Table 2, both SPEA2 and NSGA-II perform better than IBEA under all the indicators, but the differences between SPEA2 and NSGA-II are not statistically significant. In view of these results, no asseveration can be made with respect to which one of the hybridized MOEAs performs better in this context.

At this point, we advised the need of applying an *ad hoc* strategy in order to select one of the algorithms, i.e., we will play the role of a decision maker. The Table 3 shows the average of the objective's values for the biclusters found by each memetic MOEA executed with the parameters shown in Table 1. It is clear that IBEA obtains biclusters of a bigger size (on average) with respect to those obtained by SPEA2 and NSGA-II. Moreover, NSGA-II constitutes the approach that obtains the most homogeneous biclusters and SPEA2 is the one that obtains the best relation between residue and row variance. This behavior becomes more evident for the *Yeast* dataset than for the *Lymphoma* dataset. It is important to notice that, in general, biclusters with higher size have higher residue and lower row variance; whereas biclusters with small residue have sizes that tend to be smaller, independently of the row variance. The row variance is at least bigger in value than the residue in all the cases.

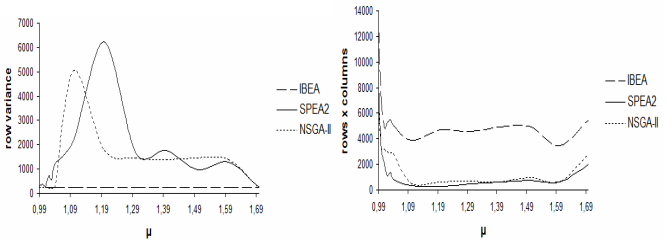


Fig. 2. Average row variance (left) and average size (right) for IBEA, NSGA-II and SPEA2 on the *Yeast* dataset when parameter varies between 0.99 and 1.7

Another characteristic of the MOEAs that can help on the selection of the method is constituted by how well the search can be oriented by means of the μ parameter on the LS. The Figure 2 shows, for the *Yeast* dataset, the average row variance (left) and the average size (right) of the biclusters obtained by each hybrid method when the μ parameter varies between 0.99 and 1.7. This threshold is related with the values of μ that have more effect over the results. As we can see, both SPEA2 and NSGA-II are well suited for being guided by the μ parameter, since as we increase the value of μ, the resulting biclusters have higher row variance in detriment of the size. In this regard, the SPEA2 is the algorithm with best performance. On the other hand, the only effect that can be observed on IBEA is the reduction of the size of the biclusters, since we can not observe any effect on the row variance. Perhaps this is due to the fact that IBEA is an Indicator-based MOEA whereas SPEA2 and NSGA-II are Pareto-based MOEAs. Therefore, a conjecture is that the small changes introduced by the μ parameter in the population in each generation are not well perceived by IBEA; probably because the concept of non-dominated solution set is not supported by the algorithm. The behavior observed on the *Lymphoma* dataset is similar.

For the comparative study of the next sub-section, we chose the memetic SPEA2 since it is more sensitive to the μ parameter than the others, whereas the average sizes of the biclusters are similar to those found by the memetic IBEA. Although IBEA can find some greater biclusters than SPEA2, it is not sensitive to the μ parameter.

All the testing has been made on a *Mobile Sempron* with 2 GB of RAM. The running time (on average) for the Yeast dataset was of 150s whereas for the *Lymphoma* dataset was of 660s. Since the execution time is mainly influenced by the LS procedure, the three MOEAs obtained these values.

Second Experimental Phase

A comparison between the memetic SPEA2 and Mitra and Banka's algorithm [3] is presented here. For this analysis, we used the results published by [3] in the paper. The parameter setups of our approach are those of the Table 1. The Table 4 shows the average results of the objective's values for the *Yeast* dataset for both approaches. The size of the largest bicluster found by each method and the coverage of the gene expression data matrix **E** are also shown. The row variance is not shown because in [3] it is not reported. As we can see, our proposal can obtain more homogeneous biclusters (on average) whereas the biclusters of Mitra and Banka's algorithm are bigger in size (on average). The largest bicluster found by the two methods is similar in

Table 4. Average objective's values for the bicluster found in the *Yeast* dataset by our memetic SPEA2 and Mitra and Banka's approach

	average rows	average columns	average residue	average size	largest bicluster size	coverage of cells
memetic SPEA2	794,59	10,37	224,31	8239,89	14602	72,50%
M&B's approach [3]	1095,43	9,29	234,87	10176,54	14828	51,34%

size. When we consider the coverage of **E**, our proposal obtains a significantly better coverage of cells with respect to Mitra and Banka's algorithm. It is important to remark that the biclusters found by our approach also include the inverted rows; therefore, the search is carried out over a doubled-size search space with regard to the other evolutionary methods for microarray biclustering found in the literature.

As regards to the *Lymphoma* dataset, in [3] the average results of the objective's values are not reported in the paper. For this dataset, they simply show the largest bicluster and the average coverage of **E**. In this regard, our proposal can find a bicluster greater than the one reported for Mitra and Banka's algorithm. This bicluster has 1009 genes, 63 conditions, a mean squared residue of 1181.06, a row variance of 1295.05, and a size of 63567; whereas the greatest bicluster that is reported in [3] is of a size of 37560. We can argue that, to the best of our knowledge, this bicluster is greater than any other bicluster found by any method reported in the existing literature. Also, the coverage of **E** achieved by our memetic SPEA2 is (on average) about 33.58% of cells; significantly better than the average of 20.96% obtained by Mitra and Banka's algorithm.

6 Conclusions

In this paper, we have introduced a general multi-objective framework for microarray biclustering hybridized with a LS procedure for finer tuning. In a first experimental phase, we have hybridized and compared three well known MOEAs (IBEA, SPEA2 and NSGA-II) based on the PISA platform, in order to establish which one obtains the best results. Since no conclusive result was obtained from this evaluation, we selected the SPEA2 since it was able to obtain relatively large biclusters with a high sensitivity to the μ parameter. Then, during a second experimental phase, we have demonstrated that the quality of the outcomes of the memetic SPEA2 outperformed the results reported by Mitra and Banka. The comparative assessment was carried out on two benchmark gene expression datasets to demonstrate the effectiveness of the proposed method.

Moreover, we provide to the biological scientists with an extra parameter to determine which biclusters they consider more relevant, giving them the possibility of adjusting the size and the row variance of the biclusters. Furthermore, the evolutionary approaches for biclustering found in the literature do not consider the inclusion of inverted rows, perhaps for efficiency reasons since the search space is duplicated. However, these inverted rows are very important because, they can be interpreted as co-regulated by receiving the opposite regulation. In this context, we have also demonstrated that it possible to take into account these "extra rows" thus improving the quality of the biclusters, without loss of efficiency.

Acknowledgments. Authors acknowledge the ANPCyT from Argentina, for Grant N°11-1652. They would also like to acknowledge SeCyT (UNS) for Grant PGI 24/ZN15.

References

1. Madeira, S., Oliveira, A.L.: Biclustering Algorithms for Biological Data Analysis: A Survey. IEEE-ACM Trans. Comput. Biol. Bioinform. 1, 24–45 (2004)
2. Cheng, Y., Church, G.M.: Biclustering of Expression Data. In: Proceedings of the 8th International Conf. on Intelligent Systems for Molecular Biology, pp. 93–103 (2000)
3. Mitra, S., Banka, H.: Multi-objective evolutionary biclustering of gene expression data. Pattern Recognit. 39, 2464–2477 (2006)
4. Divina, F., Aguilar-Ruiz, J.S.: Biclustering of Expression Data with Evolutionary Computation. IEEE Trans. Knowl. Data Eng. 18(5), 590–602 (2006)
5. Bleuler, S., Prelic, A., Zitzler, E.: An EA framework for biclustering of gene expression data. In: Proceeding of Congress on Evolutionary Computation, pp. 166–173 (2004)
6. Cho, R., et al.: A genome-wide transcriptional analysis of the mitotic cell cycle. Mol. Cell 2(1), 65–73 (1998)
7. Menges, M., Hennig, L., Gruissem, W., Murray, J.: Genome-wide gene expression in an Arabidopsis cell suspension. Plant Mol. Biol. 53(4), 423–442 (2003)
8. Laule, O., et al.: Crosstalk between cytosolic and plastidial pathways of isoprenoid bio systhesis in arabidopsis thaliana. PNAS 100(11), 6866–6871 (2003)
9. Alizadeh, A.A., et al.: Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
10. Bleuler, S., Laumanns, M., Thiele, L., Zitzler, E.: PISA - A Platform and Programming Language Independent Interface for Search Algorithms. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 494–508. Springer, Heidelberg (2003)
11. Zitzler, E., Künzli, S.: Indicator-Based Selection in Multiobjective Search. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 832–842. Springer, Heidelberg (2004)
12. Deb, K., Agraval, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000)
13. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In: Giannakoglou, Tsahalis, Periaux, Papailiou, Fogarty (eds.) Evolutionary Methods for Design, Optimisations and Control, pp. 19–26 (2002)
14. Knowles, J., Thiele, L., Zitzler, E.: A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. TIK Computer Engineering and Networks Laboratory (2005)
15. Zitzler, E., Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Trans. Evol. Comput. 3(4), 257–271 (1999)
16. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., Grunert da Fonseca, V.: Performance Assessment of Multiobjective Optimizers: An Analysis and Review. IEEE Trans. Evol. Comput. 7(2), 117–132 (2003)
17. Hansen, M., Jaszkiewicz, A.: Evaluating the quality of approximations to the non-dominated set. Technical University of Denmark (1998)
18. Conover, W.: Practical Nonparametric Statistics. John Wiley & Sons, New York (1999)

F-score with Pareto Front Analysis for Multiclass Gene Selection

Piyushkumar A. Mundra[1] and Jagath C. Rajapakse[1,2,3]

[1] Bioinformatics Research Center, School of Computer Engineering, Nanyang Technological University, Singapore
[2] Singapore-MIT Alliance, Singapore
[3] Department of Biological Engineering, Massachusetts Institute of Technology, USA
asjagath@ntu.edu.sg

Abstract. F-score is a widely used filter criteria for gene selection in multiclass cancer classification. This ranking criterion may become biased towards classes that have surplus of between-class sum of squares, resulting in inferior classification performance. To alleviate this problem, we propose to compute individual class wise between-class sum of squares with Pareto frontal analysis to rank genes. We tested our approach on four multiclass cancer gene expression datasets and the results show improvement in classification performance.

1 Introduction

DNA microarray technology has enabled us to measure gene expressions of thousands of genes simultaneously. However, due to high cost of experiments, sample sizes of gene-expression measurements remain in hundreds. Extraction of useful information from such high dimensional datasets are hindered by curse of dimensionality as well as computational instabilities. Hence, selection of relevant genes is extremely important in microarray gene-expression analysis. Various gene selection methods, with both filter and wrapper approaches, have been developed for two-class problem [1,2,3,4,5]. But as the number of classes increases, two-class gene selection methods may not be applicable and special attention is needed for gene selection considering the complexity of the problem.

Li et al. has given excellent comparative review on multiclass classification and feature selection methods [6]. F-score is a popularly applied filter approach for multiclass gene selection [7]. It is based on classical F-statistics, a generalization of T-test for two sample comparison. F-statistic presumes equal variance and to relax such an assumption, Chen et al. proposed to use Brown-Forsythe test statistic and Welch test statistic [8]. Other widely used feature selection methods include mutual information and correlation coefficients, in which, the relevance of a gene to class labels is measured by either of these scores. Based on these filter criteria, specialized algorithms like minimum redundancy maximum relevancy (MRMR) [2] and differential degree of prioritization (DDP) [3] have been proposed to reduce redundancy among top-ranked genes. In another heuristic

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 56–67, 2009.

approach, genes are ranked by using mean and standard deviation with the assumption that they have short distances from class centroids and simultaneously have small within class variations [9].

In wrapper approaches, Zhou and Tick have recently proposed multiclass support vector machine - recursive feature elimination (MSVM-RFE) method for multiclass cancer gene selection, which is an extension of two class SVM-RFE [10]. Similarly, Duan and Rajapakse proposed extension of multiple SVM-RFE for multiclass gene selection [11]. Apart from these, few evolutionary algorithms based methods have been developed for multiclass gene selection [12], [13].

As seen above, many of the multiclass gene selection methods are extensions of two class problems. The widely used F-score criterion is a ratio of the between-class sum of squares to within class sum of squares of individual genes. As this is a generalization of two class T-test, this ranking criterion will be biased towards strongly predictive genes of some classes due to surplus component of the between-class sum of squares, while ignoring the genes required to discriminate difficult classes. This may lead to poor representation of a few classes and over-representation of other classes, resulting in poor classification performances. To overcome such problem, Xuan et al. designed a two-step strategy, in which, individually discriminatory genes were identified based on one dimensional Fisher criterion, and subsequently, jointly discriminatory genes were selected based on multidimensional weighted Fisher criterion [14]. But estimating weights is a difficult problem in such an approach. To solve a similar problem, Forman proposed round robin algorithm using Infogain as a feature selection method for multi-class text classification [15]. Hero and Fleury [16] proposed a pareto-front analysis based method to rank time-series gene-expression profiles.

In this paper, we propose to compute individual classwise between-class sum of squares with Pareto-front analysis to rank genes. In the proposed approach, F-score is decomposed for an individual class in one vs all approach and treated as a muti-objective criterion. Pareto-front analysis, widely used in multi-objective evolutionary optimization, is applied to rank genes with multi-objective criteria. We test our approach on four multiclass cancer gene expression datasets and the results show significant improvement in classification accuracy. This paper is organized as follows: in following section, we present a detailed description of F-score and the proposed method with algorithms. Numerical experimental procedures and results are discussed in the next section and finally, the last section concludes the paper with a discussion.

2 Method

Let $D = \{x_{ij}\}_{n \times m}$ denotes the microarray gene expression dataset where x_{ij} is gene expression of ith gene in jth sample; Here, n represents total number of genes and m denotes number of samples. Let $x_j = (x_{1j}, x_{2j}, \ldots, x_{nj})$ be the gene expressions measured in the jth sample and the target class label of jth sample be $y_j \in \Gamma = \{1, 2, \ldots, \ell\}$, taking from ℓ different tissue classes.

F-score is based on statistical F-test and is used for filtering genes with near-constant expression across all tumour samples from important genes [7]. Intuitively, F-score will be large for a gene whose expression varies relatively small within a class compared to large variations between other classes. For gene i, F-score is computed by

$$F(i) = \frac{\sum_j \sum_\ell \delta(y_j = \ell)(\overline{x}_{i\ell} - \overline{x}_{i.})^2}{\sum_j \sum_\ell \delta(y_j = \ell)(x_{ij} - \overline{x}_{i\ell})^2} \tag{1}$$

Here, $\overline{x}_{i.}$ is average expression level of gene i across all samples, $\overline{x}_{i\ell}$ is the average expression level of gene i across samples belonging to ℓ-th class, and $\delta(.)$ denotes indicator function, equal to 1 if the arguement is true and 0 otherwise. High F-score of a gene represents high importance of that gene in classification.

As seen in Eq. 1, F-score is the ratio of between-class sum of squares to within class sum of squares. For any gene, if between-class sum of squares is very high for few classes, it may blind the discriminating effect needed for other difficult class separation. As a result, many genes will be selected for such strong classes and a very few will be top-ranked for difficult classes.

We look at this problem from a multiobjective perspective. Instead of summing up all between-class sum of square components in numerator, we take each component as one objective. After computing individual components, we apply standard Pareto-front analysis, used in multi-objective evolutionaty optimization algorithms, for ranking genes. Mathematically, proposed objective functions (i.e. class-wise F-score) are

$$\max \;\; F_\ell(i) = \frac{(\overline{x}_{i\ell} - \overline{x}_{i.})^2}{\sum_j \sum_\ell \delta(y_j = \ell)(x_{ij} - \overline{x}_{i\ell})^2}, \forall \ell \in \Gamma \tag{2}$$

In above formulation, we will have ℓ objective functions for ℓ classes. Higher F_ℓ is desired in each class for overall better classification performance. For Pareto-front analysis, we define following gene dominance definitions based on multi-objective optimization theory.

Definition 1. *A gene i is said to dominate the other gene i', if both the following conditions 1 and 2 are true:*

(1) The class-wise F-score (F_ℓ) of gene i is no worse than that of gene i' in all classes
(2) At least one of the class-wise F-score (F_ℓ) of gene i is better than that of gene i'

Definition 2. *Among a set of genes S, the non-dominated set of genes S' are those genes that are not dominated by any gene of the set S.*

After Pareto-front analysis, we obtain $\mathcal{P}$ number of ranked fronts, each front containing several number of genes. To rank genes within a front, we sum all the F-score values and sort it in descending order. The detailed method of the

Algorithm 1. F-score with Pareto-Front Analysis (F-PFA)

Begin : Gene subset $S = [1, 2, \ldots, n]$
Compute class-wise F-score (F_ℓ) for all genes in S using Eq. 2
Determine Pareto-fronts using genes non-dominated sorting algorithm
while genes in each front are ranked **do**
In each front, sum all class-wise F-score in each gene i with $F(i) = \sum_\ell F_\ell(i)$
Sort F values and rank genes in each front
end while
output : Ranked Genes

gene ranking is described in Algorithm 1. For Pareto-front analysis of given genes with their class-wise F-score, we can perform all pair-wise comparisons and determine non-dominated genes. If these genes are non-dominated by any other genes in the whole set, they forms Pareto-optimal set. In gene selection context, Pareto-optimal set represents genes which are more important for classification than genes outside this set. NSGA-II is one of the very popular algorithms to obtain Pareto-optimal set [17]. We present this algorithm in gene selection context (with minimization of $-F_\ell$ as objective), as shown in Algorithm 2.

The *for*-loop in Algorithm 2 determines how many number of genes dominates a particular gene i, and subset of genes which are dominated by gene i (If there are no genes which dominates the gene i, then gene i belongs to first Pareto-front). The second loop (*while*) is then used to separate genes in subsequent Pareto-fronts. [See [17] for further details].

It is important to note that microarray gene expression data contains thousands of genes. Due to high computational complexity of Pareto-ranking methods, it is not feasible to use all genes in the analysis. In this paper, we only use top 1000 genes ranked by F-score methods for subsequent Pareto-front analysis.

3 Experiments and Results

3.1 Data

To evaluate the performance of the proposed F-PFA method, we performed extensive experiments on four microarray gene expression datasets, namely, GCM [18], Lung [19], MLL [20], and NCI Ross [21]. These are widely used benchmark datasets to evaluate gene ranking methods with varying number of classes and genes. Except GCM dataset, there is no separate testing set available for other datasets. Hence, we divided the original dataset into separate training set and testing set in Lung, MLL, and NCI Ross datasets. Further, all datasets need pre-processing for gene expression values.

3.2 Preprocessing

All four datasets contain a large number of genes, out of which many genes have constant gene expression levels. Therefore, it becomes necessary to pre-process

Algorithm 2. Genes non-dominated sorting (modified from [17])

Begin : Gene subset S with corresponding F-scores $\{-F_\ell\}$
for each gene $i \in S$ **do**
  $Z_i = \emptyset$
  $n_i = 0$
  for each gene $i' \in S$ **do**
    if $(i \prec i')$ **then**
      $Z_i = Z_i \cup \{i'\}$
    else if $(i' \prec i)$ **then**
      $n_i = n_i + 1$
    end if
  end for
  if $n_i = 0$ **then**
    $i_{rank} = 1$
    $\mathcal{F}_1 = \mathcal{F}_1 \cup \{i\}$
  end if
end for
$r = 1$
while $\mathcal{F}_r \neq \emptyset$ **do**
  $Q = \emptyset$
  for each gene $i \in \mathcal{F}_r$ **do**
    for each gene $i' \in Z_i$ **do**
      $n_{i'} = n_{i'} - 1$
      if $n_{i'} = 0$ **then**
        $i'_{rank} = r + 1$
        $Q = Q \cup \{i'\}$
      end if
    end for
  end for
  $r = r + 1$
  $\mathcal{F}_r = Q$
end while
output : Genes separated in different Pareto fronts

data. For NCI Ross dataset, we employed pre-process dataset as given by the supplementary information of [22]. For other datasets, we employ the following steps for pre-processing:

1. Absolute gene expression values were thresolded with floor values of 100 and ceiling of 16000.
2. From training dataset, we excluded genes with absolute gene expression value change less than 100 across all training samples.
3. Transformation with log10 base.

We used above pre-processed datasets for F-score ranking and F-PFA. For classification performance evaluation, genes were further standardized to zero mean and one standard deviation in all datasets. The number of samples and genes before and after pre-processing are given in Table 1.

Table 1. Sample Sizes of Four Multiclass Gene-Expression Datasets

Dataset	# Class	# Training	# Testing	# Genes after preprocessing
GCM	14	144	54	13842
MLL	3	59	13	10848
NCI(Ross)	8	46	12	5642
Lung	5	168	35	5220

3.3 Performance Evaluation

Ranking of genes in each dataset was obtained using F-score and proposed F-PFA method. Only training data was used to rank the genes. Using the gene ranking list, we tested gene subsets starting from top ranked gene and then successively adding one gene at a time in the testing subset till the total number of genes in subset equals 200.

As seen from Table 1, the number of samples in training and testing sets are small and it can cause a peculiar problem while dividing into training and testing sets. The performance evaluation is not reliable if only one set of testing set is used. To avoid bias in performance evaluation, test errors were evaluated on bootstraped samples by merging the training and testing datasets, and then, employing stratified sampling to partition the total samples into separate training and testing sets by maintaing number of samples in each set as before. The classifier is then trained on the training set and tested on the corresponding testing set. This process is followed for 100 times and performance measure such as, test accuracy and Gorodkin Correlation Coefficient(GCC), were computed for these 100 trials. The mean and standard deviation of these performance measures are shown in Table 2 and Table 3.

In datasets with small sample size with unbalanced class distribution, it is not reliable enough to evaluate performance based on accuracy only. In such cases, Matthew's correlation coefficient (MCC) is used for binary classification. We employ GCC performance measure, which is generalization of MCC for multiclass classification problem [23], [24] .

Assuming the actual observation matrix O and prediction matrix P are two $m \times \ell$ matrices (with m number of samples and ℓ number of classes), GCC computes correlation between both matrices. An element in O_{jl} is equal to 1 if jth sample belongs to class l and 0 otherwise. Similarly in prediction matrix P_{jl}, an element is 1 if jth sample is predicted to belong to class l and 0 otherwise. GCC is defined as

$$GCC = \frac{COV(O,P)}{\sqrt{COV(O,O)}\sqrt{COV(P,P)}} \tag{3}$$

where $COV(O,P)$, $COV(O,O)$, and $COV(P,P)$ are covariances of the corresponding matrices, defined as arithmetic average of the covariance of corresponding columns of the matrices. Similar to Pearson's correlation coefficient and MCC, GCC values vary between [-1,1] range. Higher the GCC value, better the prediction is.

Table 2. Performance of F-score and F-PFA method on Various *Cancer* Datasets

Dataset	Measurement	F-score	F-PFA
GCM	Gene	**77**	96
	Accuracy	67.72 ± 4.50	$\mathbf{72.43 \pm 4.72}$
	GCC	0.66 ± 0.05	$\mathbf{0.71 \pm 0.05}$
MLL	Gene	29	**18**
	Accuracy	$\mathbf{99.20 \pm 2.31}$	99.00 ± 2.60
	GCC	$\mathbf{0.99 \pm 0.03}$	$\mathbf{0.99 \pm 0.04}$
NCIRoss	Gene	**51**	57
	Accuracy	$\mathbf{78.75 \pm 9.65}$	78.50 ± 9.03
	GCC	$\mathbf{0.77 \pm 0.11}$	$\mathbf{0.77 \pm 0.10}$
Lung	Gene	89	**18**
	Accuracy	95.34 ± 2.86	$\mathbf{96.57 \pm 2.87}$
	GCC	0.91 ± 0.05	$\mathbf{0.94 \pm 0.05}$

Table 3. Comparison of proposed method with MSVM-RFE method on top 400 genes with $\eta = 0.1$ and 4-fold cross validation

Dataset	Measurement	F-score	MSVM-RFE	F-PFA
GCM	Accuracy	72.35 ± 5.55	$\mathbf{83.25 \pm 1.48}$	77.20 ± 5.19
	GCC	0.70 ± 0.06	...	$\mathbf{0.76 \pm 0.06}$
MLL	Accuracy	98.40 ± 2.94	97.67 ± 0.83	$\mathbf{98.55 \pm 2.77}$
	GCC	0.98 ± 0.04	...	$\mathbf{0.98 \pm 0.04}$
NCIRoss	Accuracy	79.86 ± 9.70	71.31 ± 3.45	$\mathbf{81.73 \pm 9.32}$
	GCC	0.78 ± 0.11	...	$\mathbf{0.80 \pm 0.10}$
Lung	Accuracy	95.08 ± 2.67	94.9 ± 0.83	$\mathbf{95.39 \pm 2.57}$
	GCC	0.90 ± 0.05	...	$\mathbf{0.91 \pm 0.05}$

To compare performance of proposed method with the results of recently proposed multiclass SVM-RFE method [10], we merged all training and testing samples in each dataset. Four fold cross-validation accuracy was measured 100 times with all samples and 400 top ranked genes in each dataset. We only compared our results with MSVM-RFE with one vs all (OVA) SVM formulation with sensitivity parameter $\eta = 0.1$. In all testing methods, we used LIBSVM - 2.84 software [25] with OVA formulation. We employed linear SVM for testing top ranked gene subsets. Performance of linear SVM depends on sensitivity parameter η values, hence for results shown in Table 2, η was tuned from finite set $\{2^{-20}, \ldots, 2^{0}, \ldots, 2^{15}\}$.

3.4 Results

Table 2 shows maximum accuracy achieved and GCC values for 100 times random stratafied testing on all four datasets. For GCM and lung cancer datasets, classification performance were significantly improved compared to F-score method. Though results in Table 2 did not show any improvement of

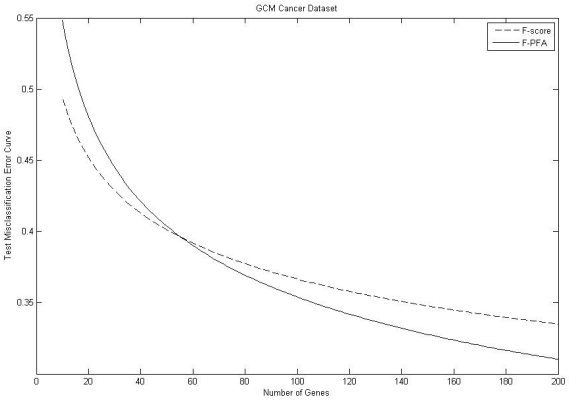


Fig. 1. Trend of misclassification error rate for F-PFA and F-score based gene ranking methods on GCM Cancer Dataset against the number of genes

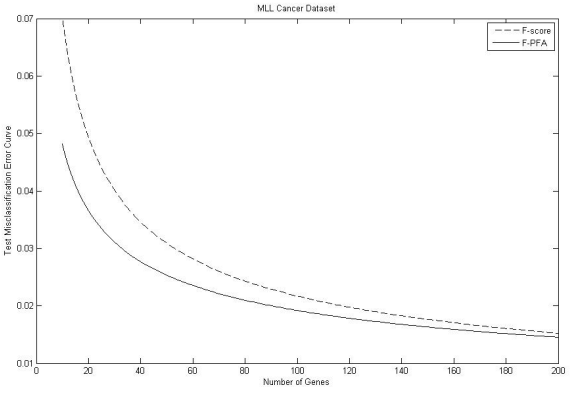


Fig. 2. Trend of misclassification error rate for F-PFA and F-score based gene ranking methods on MLL Cancer Dataset against the number of genes

performance in MLL and NCI Ross datasets, performance improves considerably with top 400 genes and four fold cross validation, as shown in Table 3. Considering the top ranked genes, the proposed method has good performance than F-score ranking in all four datasets. Number of genes required for classification are determined using "one-standard error rule" [26]. Figure 1, 2, 3, and 4 represent the test misclassification error curve in each of four dataset. As seen from Figure 2 and 4, F-PFA method needs significantly less number of genes to reach plateau in misclassification curve.

To compare with recently developed MSVM-RFE method, we performed hundred times 4-fold cross-validation to measure classification performance with top 400 genes. Performance comparison results of both the methods are shown in

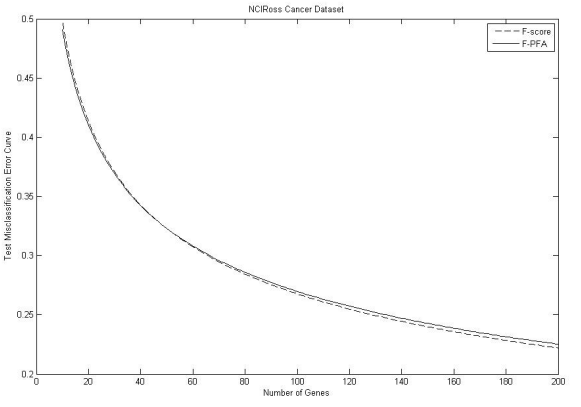


Fig. 3. Trend of misclassification error rate for F-PFA and F-score based gene ranking methods on NCI Ross Cancer Dataset against the number of genes

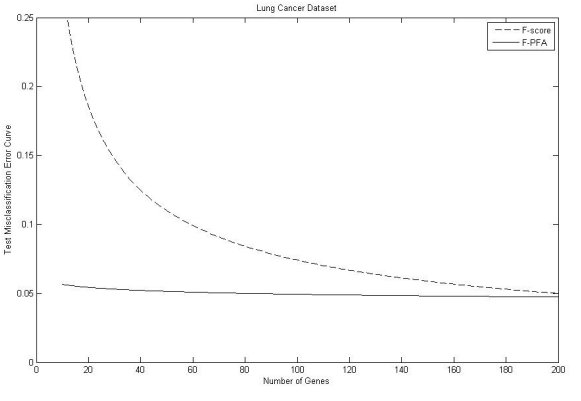


Fig. 4. Trend of misclassification error rate for F-PFA and F-score based gene ranking methods on Lung Cancer Dataset against the number of genes

Table 3. Except GCM dataset, F-PFA improves four fold cross-validation performance compare to MSVM-RFE method. It is important to note that accuracy with MSVM-RFE method were reported using 100 times four fold external cross-validation [Supplement Info, [10]], compare to simple 100 times four-fold cross-validation with F-PFA and F-score. The advantage of proposed F-PFA method is that it is computationally fast compared to MSVM-RFE, which is a wrapper method and hence, computationally very expensive.

4 Discussion and Conclusion

We presented Pareto-front analysis based F-score ranking method for gene selection in multiclass problem. Here, the hypothesis is that easy class separation

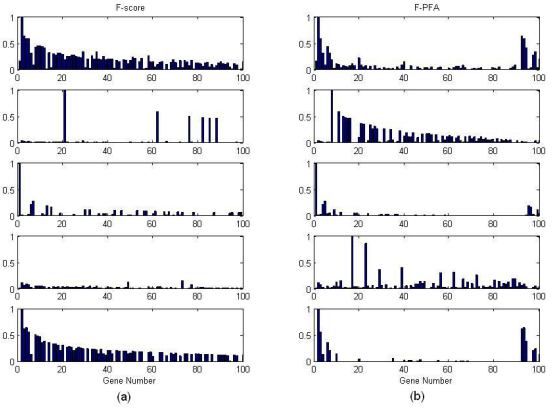


Fig. 5. (a) Bar chart of normalized objective value for each objective in Lung Cancer Dataset with F-score (b) Bar chart of normalized objective value for each objective in Lung Cancer Dataset with F-PFA

of one class from others can affect gene ranking, and may reduce overall classification performance as genes responsible for difficultly separable classes may not be represented adequately.

Above phenomenon is depicted in Fig. 5. Here, we plotted normalized objective value of top 100 genes ranked by F-score and F-PFA method in Lung cancer dataset. As seen from the figure, there are no genes representing class 4 with standard F-score based ranking. At the same time, class 1 and 5 are over-represented. On the other hand, similar plots for proposed F-PFA method shows equal representation for every class. In fact, there is reduction in number of genes for class 1 and 5 while an increase in number of genes for class 2 and 4. This resulted in significant improvement in the classification performance.

As discussed in [17], non-dominated sorting with NSGA-II has computational complexity of $O\left(\ell n^2\right)$, where ℓ is number of objectives and n is population size. In gene ranking context, n is number of genes in dataset. Though another fast algorithm with computational complexity of $O\left(n \log^{\ell-1} n\right)$ had been proposed [27], because of very large number of genes in microarray datasets, the computational complexity still remains very high with proposed method. In present work, we have applied Pareto-front analysis on only top 1000 genes ranked from standard F-score.

In proposed method, increase in number of classes, i.e. number of objectives, may result in large number of genes in first front itself. To avoid it, we used one-vs-all approach for individual class wise objective computation. If there are ℓ classes, one-vs-all approach will result in ℓ objective values, compare to $\ell(\ell-1)/2$ objective values with one-vs-one approach. Due to this strategy, we were able to reduce number of genes in each front.

In conclusion, we have presented a new Pareto-front based F-score method for gene ranking in multiclass problem. This method was developed by treating

class-wise F-score as multiple objectives and comparing genes with Pareto analysis. Though the performance was not consistent across all datasets, it seems to select less number of genes and/or performs better on most datasets.

References

1. Inza, I., Larranaga, P., Blanco, R., Cerrolaza, A.: Filter versus wrapper gene selection approaches in DNA microarray domains. Artificial Intelligence Medicine 31, 91–103 (2004)
2. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. J. Bioinformatics Computational Biology 3, 185–205 (2005)
3. Ooi, C., Chetty, M., Teng, S.: Differential prioritization between relevance and redundancy in correlation-based feature selection techniques for multiclass gene expression data. BMC Bioinformatics 7, 320–339 (2006)
4. Kai-Bo, D., Rajapakse, J., Wang, H., Azuaje, F.: Multiple SVM-RFE for gene selection in cancer classification with expression data. IEEE Trans. Nanobioscience 4, 228–234 (2005)
5. Mundra, P., Rajapakse, J.: SVM-RFE with relevancy and redundancy criteria for gene selection. In: Rajapakse, J.C., Schmidt, B., Volkert, L.G. (eds.) PRIB 2007. LNCS (LNBI), vol. 4774, pp. 242–252. Springer, Heidelberg (2007)
6. Li, T., Zhang, C., Ogihara, M.: A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. Bioinformatics 20(15), 2429–2437 (2004)
7. Dudoit, S., Fridlyand, J., Speed, T.P.: Comparison of discrimination methods for the classification of tumors using gene expression data. J. American Statistical Association 97(457), 77–86 (2002)
8. Chen, D., Liu, Z., Ma, X., Hua, D.: Selecting genes by test statistics. J. Biomedicine and Biotechnology 2, 132–138 (2005)
9. Cho, J.-H., Lee, D., Park, J.H., Lee, I.-B.: New gene selection method for classification of cancer subtypes considering within-class variation. FEBS Letters 551, 3–7 (2003)
10. Zhou, X., Tuck, T.P.: MSVM-RFE: Extensions of SVM-RFE for multiclass gene selection on dna microarray data. Bioinformatics 23(9), 1106–1114 (2007)
11. Duan, K.B., Rajapakse, J., Nguyen, M.: One-versus-one and one-versus-all multiclass SVM-RFE for gene selection in cancer classification. In: Marchiori, E., Moore, J.H., Rajapakse, J.C. (eds.) EvoBIO 2007. LNCS, vol. 4447, pp. 47–56. Springer, Heidelberg (2007)
12. Jirapech-Umpai, T., Aitken, S.: Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes. BMC Bioinformatics 6, 148–158 (2005)
13. Ooi, C.H., Tan, P.: Genetic algorithms applied to multi-class prediction for the analysis of gene expression data. Bioinformatics 19(1), 37–44 (2003)
14. Xuan, J., Wang, Y., Dong, Y., Feng, Y., et al.: Gene selection for multiclass prediction by weighted fisher criterion. EURASIP J. Bioinformatics and Systems Biology 2007(article id 64628) (2007)
15. Forman, G.: A pitfall and solution in multi-class feature selection for text classification. In: Proceedings of the twenty-first international conference on Machine learning (2004)

16. Hero, A., Fleury, G.: Pareto-optimal methods for gene ranking. J. VLSI Signal Processing 38, 259–275 (2004)
17. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evolutionary Computation 6(2), 182–197 (2002)
18. Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S.: Others: Multiclass cancer diagnosis using tumor gene expression signatures. Proceedings of the National Academy of Sciences 98(26), 15149–15154 (2001)
19. Bhattacharjee, A., Richards, W., Staunton, J., Li, C.: Others: Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses. Proceedings of the National Academy of Sciences 98(24), 13790–13795 (2001)
20. Armstrong, S., Staunton, J., Silverman, L., Pieters, R., et al.: MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. Nature Genetics 30(1), 41–47 (2002)
21. Ross, D.T., Scherf, U., Eisen, M.B., Perou, C., et al.: Systematic variation in gene expression patterns in human cancer cell. Nature Genetics 24(3), 227–235 (2000)
22. Culhane, A., Perriere, G., Higgins, D.: Cross-platform comparison and visualisation of gene expression data using co-inertia analysis. BMC Bioinformatics 4(1), 59 (2003)
23. Gorodkin, J.: Comparing two K-category assignment by a K-category correlation coefficient. Computational Biology and Chemistry 28, 367–374 (2004)
24. Liu, J., Kang, S., Tang, C., Ellis, L.B., Li, T.: Meta-prediction of protein subcellular localization with reduced voting. Nucleic Acid Research 35(15), e96 (2007)
25. Chang, C., Lin, C.: Libsvm: A library for support vector machines (2001), www.csie.ntu.edu.tw/~cjlin/libsvm
26. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, Heidelberg (2001)
27. Jensen, M.: Reducing run-time complexity of multiobjective EAs: The NSGA-II and other algorithms. IEEE Trans. Evolutionary Computation 7(5), 503–515 (2003)

A Hierarchical Classification Ant Colony Algorithm for Predicting Gene Ontology Terms

Fernando E. B. Otero, Alex A. Freitas, and Colin G. Johnson

Computing Laboratory, University of Kent, Canterbury, UK
{febo2,A.A.Freitas,C.G.Johnson}@kent.ac.uk

Abstract. This paper proposes a novel Ant Colony Optimisation algorithm for the hierarchical problem of predicting protein functions using the Gene Ontology (GO). The GO structure represents a challenging case of hierarchical classification, since its terms are organised in a direct acyclic graph fashion where a term can have more than one parent — in contrast to only one parent in tree structures. The proposed method discovers an ordered list of classification rules which is able to predict all GO terms independently of their level. We have compared the proposed method against a baseline method, which consists of training classifiers for each GO terms individually, in five different ion-channel data sets and the results obtained are promising.

Keywords: Hierarchical classification, ant colony optimisation, protein function prediction.

1 Introduction

The large amount of uncharacterised protein data available for analysis has lead to an increased interest in computational methods to support the investigation of the role of proteins in an organism. Protein classification schemes, such as the Gene Ontology [1] are organised in a hierarchical structure, allowing the annotation of protein at different levels of detail. In a hierarchical protein classification scheme, nodes near the root of the hierarchy represent more general functions while nodes near the leaves of the hierarchy represent more specific functions. The hierarchy also defines parent-child relationships between nodes were the child node is a specialisation of the parent node. From a data mining perspective, hierarchical classification is more challenging than single-level 'flat classification' [2]. Firstly, it is generally more difficult to discriminate between classes represented by leaf nodes than more general classes represented by internal nodes, since the number of examples per leaf node tends to be smaller compared to internal nodes. Secondly, an example may have more than one class predicted depending of its level in the class hierarchy and these predictions must satisfy hierarchical parent-child relationships.

This paper focuses on hierarchical protein function prediction using the Gene Ontology (GO) 'molecular function' domain. Note that the GO has a complex hierarchical organisation, where nodes are arranged in a directed acyclic graph

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 68–79, 2009.

(DAG) structure and a particular node can have more than one parent — in contrast to only one parent in tree structures. We propose a new Ant Colony Optimisation (ACO) [3] classification algorithm, named *h*Ant-Miner (hierarchical classification Ant-Miner), for the hierarchical problem of protein function prediction using the GO structure. The proposed method discovers classification rules that predict functions at all levels of the GO hierarchy, and at the same time, which are consistent with the parent-child hierarchical relationships.

The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 describes the proposed hierarchical classification method using ACO. Section 4 describes the methodology used for the data preparation. Section 5 presents the computational results of the proposed method. Finally, Section 6 draws the conclusion of this paper and discuss future research directions.

2 Related Work on Gene Ontology Term Prediction

Much work on hierarchical classification of protein functions using the Gene Ontology has been focused on training a classifier for each GO term independently, using the GO hierarchy to determine positive and negative examples associated with each classifier [4], [5], [6]. Predicting each GO term individually has several disadvantages [7]. Firstly, it is slower since a classifier need to be trained n times (where n is the number of GO terms in the GO being predicted). Secondly, some GO terms could potentially have few positive examples in contrast to a much greater number of negative examples, particularly GO terms at deeper levels of the hierarchy. Many classifiers have problems with imbalanced class distributions [8]. Thirdly, individual predictions can lead to inconsistent hierarchical predictions, since parent-child relationships between GO terms are not imposed automatically during the training. However, more elaborate approaches can correct the individual predictions in order to satisfy hierarchical relationships — e.g. a Bayesian network is used to correct the inconsistent predictions of a set of SVM classifiers in [9]. Fourthly, the discovered knowledge identifies relationships between predictor attributes and each GO term individually, rather than relationships between predictor attributes and the GO hierarchy as a whole, which could give more insight about the data.

In order to avoid the aforementioned disadvantages of dealing with each GO term individually, a few authors have proposed classification methods that discover a single global model which is able to predict GO terms at any level of the hierarchy. Kiritchenko et al. [10] present an approach where the hierarchical problem is cast as a multi-label problem by expanding the label set (GO terms) of an example with all ancestor labels (ancestor GO terms). Then, a multi-label classifier is applied to the modified data set. For some examples, there is still need for a post-processing step to resolve inconsistencies in the GO terms predicted. Clare et al. [11] presented an adapted version of C4.5, which is able to deal with all GO term at the same time. They focused on discovering only a subset of very good rules for human analysis, rather than building a complete classification model for classifying the whole data set.

3 Proposed Method

The target problem of the proposed *h*Ant-Miner method is the discovery of hierarchical classification rules in the form *IF antecedent THEN consequent*. The antecedent of a rule is composed by a conjunction of predictor attribute conditions (e.g. LENGTH > 25 AND IPR00023 = 'yes') while the consequent of a rule is composed by set of class labels (GO terms) in potentially different levels of the GO hierarchy (e.g. GO:0005216, GO:0005244 — where GO:0005244 is a subclass of GO:0005216). *IF-THEN* classification rules have the advantage of being intuitively comprehensible to biologists. *h*Ant-Miner divides the rule construction process into two different ant colonies, one colony for creating rule antecedents and one colony for creating rule consequents, which work in a cooperative fashion. Due to this paper's size restrictions this section assumes the reader is familiar with standard Ant Colony Optimisation (ACO) algorithms [3].

In order to discover a list of classification rules, a sequential covering approach is employed to cover all (or almost all) training examples. Algorithm 1 presents a high-level pseudo-code of the sequential covering procedure. The procedure starts with an empty rule list (*while* loop) and adds a new rule while the number of uncovered training examples is greater than a user-specified maximum value (*MaxUncoveredCases* parameter). At each iteration, a rule is created by an ACO procedure (*repeat-until* loop). Given that a rule is represented by trails in two different construction graphs, antecedent and consequent, two separated colonies are involved in the rule construction procedure. Ants in the antecedent colony create trails on the antecedent construction graph while ants in the consequent colony create trails on the consequent construction graph. In order to create a rule, an ant from the antecedent colony is paired with an ant from the consequent colony, so that the construction of a rule is synchronized between the two ant colonies. Therefore, it is a requirement that both colonies have the same number of ants (*ColonySize* parameter). The antecedent and

Algorithm 1. High level pseudo-code of the sequential covering procedure.

```
begin
    training_set ← all training examples;
    rule_list ← ∅;
    while |training_set| > max_uncovered_examples do
        rule_best ← ∅;
        i ← 1;
        repeat // ACO procedure
            rule_i ← CreateRule(); // use separate ant colonies for antecedent and
            consequent
            Prune(rule_i);
            UpdatePheromones(rule_i);
            if Q(rule_i) > Q(rule_best) then
                rule_best ← rule_i;
            end
            i ← i + 1;
        until i ≥ max_number_iterations OR rule convergence;
        rule_list ← rule_list + rule_best;
        training_set ← training_set − Covered(rule_best, training_set);
    end
end
```

consequent trails are created by probabilistically choosing a vertex to be added to the current trail (antecedent or consequent) based on the values of the amount of pheromone (τ) and a problem-dependent heuristic information (η) associated with vertices. There is a restriction that the antecedent of the rule must cover at least a user-defined minimum number of examples (*MinCasesPerRule* parameter), to avoid overfitting. Once the rule construction procedure has finished, the rule constructed by the ants is pruned to remove irrelevant terms from the rule antecedent and consequent.Then, pheromone levels are updated using a user-defined number of best rules (*UpdateSize* parameter) and the best-so-far rule is stored. The rule construction procedure is repeated until a user specified number of iterations has been reached (*MaxIterations* parameter), or the best-so-far rule is exactly the same in a predefined number of previous iterations (*ConvergenceTest* parameter). The best-so-far rule found, based on a quality measure Q, is added to the rule list and the covered training examples (examples that satisfy the rule's antecedent conditions) are removed from the training set.

The proposed hAnt-Miner is an extension of the 'flat classification' Ant-Miner [12] in several important ways, as follows. Firstly, it uses two separate ant colonies for constructing the antecedent and the consequent of a rule. Secondly, it uses a hierarchical classification rule evaluation measure to guide pheromone updating. Thirdly, it uses a new rule pruning procedure. Fourthly, it uses heuristic functions adapted for hierarchical classification.

3.1 Construction Graphs

Antecedent Construction Graph. Given a set of nominal attributes $\mathcal{X} = \{x_1, \ldots, x_n\}$, where the domain of each nominal attribute x_i is a set of values $\mathcal{V}_i = \{v_{i1}, \ldots, v_{in}\}$, and a set of continuous attributes $\mathcal{Y} = \{y_1, \ldots, y_n\}$, the antecedent construction graph is defined as follows. For each nominal attribute x_i and value v_{ij} (where v_{ij} is the j-th value belonging to the domain of x_i) a vertex is added to the graph representing the term $(x_i = v_{ij})$. For each continuous attribute y_i a vertex is added to the graph representing the continuous attribute y_i. Since continuous attribute vertices do not represent a complete term (condition) to be added to a rule, when an ant visits a continuous attribute vertex, a threshold value is selected to create a term using '$<$' or '$\geq$' relational operators (e.g. $y_i < value$). The selection of this value is deterministic and incorporates task-specific knowledge, increasing the effectiveness of the algorithm [13].

Then, vertices representing an attribute term (nominal or continuous) are connected to every other vertex referring to another attribute term, with the restriction that there are no edges between nominal attribute vertices referring to the same attribute (to avoid terms such as 'IPR00023 = yes' and 'IPR00023 = no' being included in the same rule). As a result, attribute term vertices are almost fully-connected. In addition, a dummy vertex '*start*' is added and unidirectionally connected to all vertices in the construction graph. This vertex represents the starting point for creating trails.

Consequent Construction Graph. Since the class labels are hierarchically structured as a directed acyclic graph (DAG), this structure can be directly used

to represent the consequent construction graph as follows. For each class label $l_i \in \mathcal{L}$, where $\mathcal{L}$ is the hierarchy of class labels, a vertex is added to the graph. Subsequently, for every child vertex l_j of l_i where $l_j, l_i \in \mathcal{L}$, a directed connection from l_i to l_j is added to the graph. As a result, the consequent construction graph is a DAG, which is exactly the DAG of classes of the target problem, containing all classes and all parent-child class relationships in the target problem. Ants traverse the graph from the root vertex towards a leaf vertex. A created trail represents a set of predicted class labels, consistent with the hierarchy (satisfying parent-child relationships).

3.2 Rule Evaluation

Since the target problem of *h*Ant-Miner is the discovery of hierarchical classification rules, a variation of the hierarchical measure proposed in [10] is used to evaluate rules constructed by ants. The measure is a combination of both precision and recall hierarchical measures, and it takes into account the fact that an example belongs not only to its more specific class label, but also to all ancestor class labels according to the class hierarchy (except the root class label, since all examples trivially belong to the root class label by default).

As discussed earlier, the consequent of a rule is represented by a complete trail from the root class vertex to a leaf class vertex. In DAG structures, multiple paths between a given pair of class labels can exist. Therefore, immediately after an ant finishes building the consequent for rule r, the set of predicted class labels P_r of rule r is extended with the corresponding ancestor labels ($P_r{}'$) as

$$P_r{}' = P_r \cup \{\cup_{l_i \in P_r} Ancestors(l_i)\} - l_{root}\,, \tag{1}$$

where $Ancestors(l_i)$ corresponds to all ancestor class labels of the class label l_i and l_{root} is the root class label of the hierarchy. The hierarchical macro-averaged measures of precision (hP) and recall (hR) are computed as

$$hP = \frac{\sum_{i \in S_r} \frac{|T_i \cap P_r{}'|}{|P_r{}'|}}{|S_r|} \qquad hR = \frac{\sum_{i \in S_r} \frac{|T_i \cap P_r{}'|}{|T_i|}}{|S_r|}\,, \tag{2}$$

where S_r is the set of all examples covered by (satisfying the rule antecedent of) rule r and T_i is the set of true class labels of the i-th example. The hierarchical precision (hP) is the average number of true class labels that are predicted by rule r divided by the total number of predicted class labels across the examples covered by rule r. The hierarchical recall (hR) is the average number of true class labels that are predicted by rule r across the examples covered by rule r divided by the total number of true class labels which should have been predicted across the examples covered by rule r.

The rule quality measure Q is defined as a combination of the hP and hR measures, equivalent to the hierarchical F-measure, given by Equation (3)

$$Q = hF = \frac{2 \cdot hP \cdot hR}{hP + hR}\,. \tag{3}$$

3.3 Rule Pruning

The rule pruning procedure aims at improving the rule quality by removing irrelevant terms that might have been added during the rule construction process and it is applied as soon as the rule construction is completed. Recall that a rule is composed by antecedent and consequent parts, which in turn are represented by different ant trails that might contain irrelevant vertices.

A rule undergoes the pruning procedure as follows. At the first step, the quality of the rule is computed using the quality measure Q as given by Equation (3). In the second step, the rule is submitted to an iterative removal of the last term added to its antecedent while the quality of the rule is improved. At each iteration, the consequent of a candidate rule is also submitted to an iterative removal of the last added class label in an attempt to improve the generalization behaviour of the candidate rule. Note that, for the purpose of both these iterative removal procedures, the terms and classes (in the antecedent and consequent, respectively) are considered as an ordered list, and terms and classes are removed in an order inverse to the order in which they were added to the rule.

Algorithm 2 describes the rule pruning. Let $rule_r$ be the rule undergoing the pruning procedure and q_r be the quality measure of $rule_r$. At each iteration of the outer repeat loop in Algorithm 2, a candidate rule $rule_i$ is created by removing the last term of the antecedent of $rule_r$ and its quality measure q_i is computed. Subsequently, j $(0 < j < |rule_i.consequent|)$ candidate rules are sequentially created by removing the last j class label(s) of the consequent of $rule_i$. This is implemented by the inner repeat loop in Algorithm 2. If the quality measure of a $rule_j$ is higher than q_i, $rule_i$ is substituted by $rule_j$. Finally, $rule_i$ substitutes $rule_r$ if $q_r \leq q_i$, completing an iteration of the pruning procedure. This procedure is repeated until $rule_r$ has just one term left on its antecedent or a candidate rule $rule_i$ does not improve the quality over $rule_r$ (i.e. $q_r > q_i$).

Algorithm 2. Rule pruning procedure pseudo-code.

```
begin
    rule_best ← rule;
    q_best ← Q(rule_best);
    repeat
        antecedent ← rule_best.antecedent − last_term(rule_best.antecedent);
        rule_i ← antecedent + rule_best.consequent;
        q_i ← Q(rule_i);
        consequent_j ← rule_best.consequent;
        repeat
            consequent_j ← consequent_j − last_class(consequent_j);
            rule_j ← antecedent + consequent_j;
            if (Q(rule_j) > q_i) then
                rule_i ← rule_j;
                q_i ← Q(rule_j);
            end
        until |consequent_j| = 1 ;
        if (q_i ≥ q_best) then
            rule_best ← rule_i;
            q_best ← q_i;
        end
    until q_i < q_best OR |rule_best.antecedent| = 1 ;
end
```

3.4 Pheromone Trails

Pheromone Initialisation. In order to reinforce trails followed by ants that constructed good rules, pheromone values are associated with edges in the antecedent and consequent construction graphs. For each vertex i of both antecedent and consequent construction graphs, the initial amount of pheromone deposited at each edge is inversely proportional to the number of edges originating at vertex i, computed as

$$\tau_{edge_{ij}}(t=0) = \frac{1}{|E_i|}\,, \tag{4}$$

where E_i is the set of edges originating at vertex i, $edge_{ij}$ is the edge that connects vertex i to its j-th neighbour vertex and t is the time index. As a result of Equation (4), the same amount of pheromone is initially associated with every $edge_{ij}$ coming out from vertex i.

Pheromone Updating. The pheromone trails followed by ants are updated based on the quality of the rule that they represent, which in turn guides future ants towards better trails. Since a rule is composed by antecedent and consequent trails, the pheromone update procedure is divided in two steps.

In the first step, the trail that represents the antecedent of a rule r is updated as follows. Starting from the dummy 'start' vertex (0-th vertex), the pheromone value of the edge that connects the i-th vertex to the $(i+1)$-th vertex $(0 \leq i < |rule.antecedent|)$ is incremented according to

$$\tau_{edge_{ij}}(t+1) = \tau_{edge_{ij}}(t) + \tau_{edge_{ij}}(t) \cdot q_r\,, \tag{5}$$

where i and j are the i-th and j-th vertices of an edge from i to j in the trail being updated ($edge_{ij}$) and q_r is the quality measure of rule r — Equation (3).

In the second step, the pheromone value of every edge of the consequent of rule r that connects the i-th vertex to the $(i+1)$-th vertex $(0 < i < |rule.consequent|)$ is incremented according to Equation (5). Note that, before computing the rule quality, the consequent is expanded to include all ancestor class labels of the class labels originally added to the rule's consequent by an ant, since there can be multiple paths between class labels, as detailed in subsection 3.2. However, during pheromone updating, only the actual trail that was followed to create the original consequent is updated. This avoids reinforcing trails that did not directly contribute to the consequent construction.

Pheromone Evaporation. This is implemented by normalizing the pheromone values of edges of each construction graph G (antecedent and consequent). The normalization procedure indirectly decreases the pheromone of unused edges, since the pheromone of used edges has been increased by Equation (5). This normalization is given by

$$\tau_{edge_{ij}} = \frac{\tau_{edge_{ij}}}{\sum\limits_{\tau_{edge_{ij}} \in G} \tau_{edge_{ij}}}\,. \tag{6}$$

3.5 Heuristic Functions

Antecedent Heuristic Function. The heuristic function used in the antecedent construction graph is based on information theory, more specifically, it involves a measure of the entropy associated with each term (vertex) of the graph. In the case of nominal attributes, where a term has the form $(x_i = v_{ij})$, the entropy for the term is computed as

$$entropy(term_{x_i=v_{ij}}) = \sum_{k=1}^{|\mathcal{L}|} -p(l_k \,|\, term_{x_i=v_{ij}}) \cdot \log_2 p(l_k \,|\, term_{x_i=v_{ij}}) \,, \qquad (7)$$

where $p(l_k \,|\, term_{(x_i=v_{ij})})$ is the empirical probability of observing class label l_k conditional on having observed $x_i = v_{ij}$ (attribute x_i having the specific value v_{ij}) and $|\mathcal{L}|$ is the total number of class labels. The entropy is a measure of the impurity in a collection of examples, hence higher entropy values correspond to more uniformly distributed classes and smaller predictive power for the term in question. Equation (7) is a direct extension of the heuristic function of the original Ant-Miner [12] (for 'flat classification') into the problem of hierarchical classification.

In the case of continuous attributes, where a vertex represents just an attribute (and not an attribute-value pair), a threshold value v is chosen to dynamically partition the continuous attribute y_i into two intervals: $y_i < v$ and $y_i \geq v$. *h*Ant-Miner chooses the threshold value v that minimizes the entropy of the partition, given by

$$entropy(y_i, v) = \frac{|S_{y_i<v}|}{|S|} \cdot entropy(y_i < v) + \frac{|S_{y_i \geq v}|}{|S|} \cdot entropy(y_i \geq v) \,, \qquad (8)$$

where $|S_{y_i<v}|$ is the total number of examples in the partition $y_i < v$ (partition of training examples where the attribute y_i has a value less than v), $|S_{y_i \geq v}|$ is the total number of examples in the partition $y_i \geq v$, $|S|$ is the total number of training examples, and $entropy(y_i < v)$ and $entropy(y_i \geq v)$ are the entropy of the terms represented by $(y_i < v)$ and $(y_i \geq v)$ as given by Equation (7).

After the selection of the threshold v_{best}, the entropy of the term representing the continuous attribute y_i corresponds to the minimum entropy value of the two partitions and it is defined as

$$entropy(term_{y_i}) = min\,(entropy(y_i < v_{best}), entropy(y_i \geq v_{best})) \,. \qquad (9)$$

Equations (8) and (9) are derived from the Ant-Miner version for coping with continuous attributes, described in [13]. In this current paper the heuristic function is straightforwardly extended to hierarchical classification as follows. Since the entropy of the ith-term (nominal or continuous) of the antecedent construction graph varies in the range $0 \leq entropy(term_i) \leq \log_2(|\mathcal{L}|-1)$ (where $|\mathcal{L}| - 1$ is the number of class labels in the class hierarchy without considering the root class label) and lower entropy values are preferred over higher values, the heuristic function is computed as

$$\eta_{term_i} = \log_2(|\mathcal{L}| - 1) - entropy(term_i) \,, \quad \forall\, term_i \in G_A \,, \qquad (10)$$

where $term_i$ is the ith-term of the antecedent construction graph G_A. Equation (10) will give a higher probability of being selected to terms with lower entropy values, which corresponds to terms with higher predictive power.

Consequent Heuristic Function. The heuristic function used in the consequent construction graph is based on the frequency of training examples for each class label of the hierarchy, given by

$$\eta_{l_i} = |TR_{l_i}|, \quad \forall\ l_i \in G_C\ , \tag{11}$$

where $|TR_{l_i}|$ is the number of training examples that belong to class label l_i and G_C is consequent construction graph. Note that the heuristic function has a bias towards class labels that have a greater number of examples, which therefore will initially favour the discovery of rules with these class labels in the consequent. However, due to the use of a sequential covering procedure, rules predicting less frequent classes will be eventually discovered as well.

4 Bioinformatics Data Preparation

In order to evaluate the proposed method, we have created five data sets involving ion-channel protein functions. Ion channel proteins are present in all living cells and they form a pore across the cell membrane [14]. The function of ion channels is to allow specific inorganic ions (e.g. Na^+, K^+, Ca^{2+}, Cl^-) to cross the cell membrane. They play an essential role in many cell functions, such as in functions related to the nervous system, muscle contraction and T-cell activation.

The selection of the protein examples was divided into three steps. In the first step we selected a subset of the Gene Ontology hierarchy to represent the hierarchical classes to be predicted. As we focus on ion channel proteins, all the ancestor and descendant terms of the GO:0005216 (*ion channel activity*) term were selected. In the second step, we retrieved protein interaction data from the IntAct database (release 15/12/2007). Records with database cross-references to the GO terms selected in the previous step were retrieved. Since many GO terms (classes) selected in the previous step did not have a reasonable number of proteins associated with them, we discarded GO terms with less than 10 protein examples. In the third step, for each protein example retrieved in the previous step we selected the amino acid sequence and InterPro pattern references from the UniProt database (release 12.0), using the database cross-reference to UniProt found in IntAct protein records. We ended up with 147 protein examples involving 17 GO terms, which were used to create three different data sets. The first data set ('DS1 AA') used the amino acid composition information as predictor attributes, consisting of the percentage of the sequence composition relative to each of the 20 different amino acids. The second data set ('DS1 InterPro') used the InterPro pattern information as predictor attributes, consisting of boolean attributes representing the presence or absence of a particular InterPro pattern. The third data set ('DS1 IntAct') used the IntAct protein interaction data as predictor attributes, consisting of boolean attributes representing the presence or absence of a particular interaction.

Using a similar approach, without the restriction of selecting proteins with protein interaction data available, we increased the number of proteins to 359 examples to create two additional data sets. The first data set ('DS2 AA') used the amino acid composition information as predictor attributes. The second data set ('DS2 InterPro') used the InterPro pattern information as predictor attributes.

5 Computational Results

The proposed *h*Ant-Miner method was compared against a baseline approach, which consists of training a classifier for each GO term of the hierarchy individually. The J48 classifier (Weka [15] implementation of the well-known C4.5 decision tree algorithm [16]) was chosen in this approach. In this case, the GO hierarchy was used to determine the set of positive/negative examples for every individual classifier as follows. For each classifier associated with a particular GO term (the classifier which predicts if an example belongs or not to the particular GO term), the set of positive examples consists of all examples that belong to the GO term in question (as the most specific GO term as an ancestor of their most specific GO term); the set of negative examples consists of all the remaining training examples. After training the individual classifiers, a test example is classified in a top-down fashion. First, the example is classified only by the child classifiers of the root GO term. For each child classifier, if the classifier predicts the positive class (the example is predicted to belong to the GO term associated with the classifier), then the example is 'pushed downwards' and classified by its children classifiers. This procedure goes on until a classifier does not predict the GO term (the example is predicted as a negative example) or when a leaf classifier is reached. At the end of this procedure, the set of predicted class labels is consistent with the GO hierarchy.

We compare the performance of *h*Ant-Miner and J48 in terms of the hierarchical measures of precision, recall and F-measure, since standard classification accuracy measures are not suitable for hierarchical classification (i.e. they do not account for misclassification errors at different levels of the hierarchy). The hierarchical measures of precision, recall and F-measure are defined in Equations (2) and (3) respectively, with the difference that all test examples are considered as we are evaluating a classification model and not a rule (which is the case in subsection 3.2). The experiments were conducted running the well-known 10-fold cross-validation procedure [15] and the results are reported as average values with standard deviation computed over the 10 different iterations. In all experiments, the parameters of *h*Ant-Miner were set to: '$ColonySize = 20$', '$MinCasesPerRule = 5$', '$MaxUncoveredCases = 10$', '$ConvergenceTest = 10$', '$MaxIterations = 500$' and '$UpdateSize = 1$'. We have made no attempt to optimise these parameters for the data sets used in the experiments. The results comparing the proposed *h*Ant-Miner method against J48 are shown in Table 1.

Overall *h*Ant-Miner achieved better results than J48 in our set of experiments. J48 was significantly outperformed (according to a Student's t-test — see Table 1) in two out of five data sets, namely 'DS1 InterPro' and 'DS1 IntAct'. These data sets can be considered 'difficult' based on their small size (147 proteins)

Table 1. Hierarchical measures of precision (hR), recall (hR) and F-measure (hF) values (*mean* ± *standard deviation*) obtained with the 10-fold cross-validation procedure in the five data sets. An entry in the 'hF' column is shown in bold if the hierarchical F-measure value obtained by one of the methods was significantly greater than the other method — according to a two-tailed Student's t-test with 99% confidence.

	J48 (top-down)		
	hP	hR	hF
DS1 AA	0.73 ± 0.04	0.55 ± 0.03	0.63 ± 0.03
DS1 InterPro	0.69 ± 0.04	0.68 ± 0.05	0.69 ± 0.04
DS1 IntAct	0.69 ± 0.03	0.37 ± 0.04	0.47 ± 0.03
DS2 AA	0.71 ± 0.02	0.61 ± 0.02	0.65 ± 0.02
DS2 InterPro	0.91 ± 0.01	0.84 ± 0.02	**0.87 ± 0.01**
	*h*Ant-Miner		
	hP	hR	hF
DS1 AA	0.56 ± 0.06	0.55 ± 0.06	0.56 ± 0.06
DS1 InterPro	0.82 ± 0.04	0.81 ± 0.04	**0.81 ± 0.04**
DS1 IntAct	0.77 ± 0.04	0.54 ± 0.03	**0.63 ± 0.03**
DS2 AA	0.63 ± 0.02	0.59 ± 0.02	0.61 ± 0.01
DS2 InterPro	0.83 ± 0.01	0.75 ± 0.01	0.79 ± 0.01

and distribution of examples in the GO hierarchy (GO terms at deeper levels of the hierarchy have few examples). Therefore, the poor performance of J48 could be the result of the problem that there are many more negative examples than positive examples for each GO node, particularly at deeper level in the GO hierarchy, which shows that *h*Ant-Miner is more robust than J48 when dealing with unbalanced hierarchical class distributions. This problem was not observed in the experiments concerning the data sets with a greater number of protein examples, were J48 significantly outperformed *h*Ant-Miner in the 'DS2 InterPro' data set. In the remaining two data sets, namely 'DS1 AA' and 'DS2 AA', there were no significant difference between both methods.

6 Conclusion

This paper has presented a new Ant Colony Optimisation algorithm, named *h*Ant-Miner, for the hierarchical classification problem of predicting protein functions using the Gene Ontology (GO). The proposed *h*Ant-Miner discovers a single global classification model in the form of an ordered list of *IF-THEN* classification rules which can predict GO terms at all levels of the GO hierarchy, satisfying the parent-child relationships between GO terms. Experiments comparing *h*Ant-Miner with a baseline method based on J48, where one classifier is trained individually for each GO term of the hierarchy, have shown positive results: *h*Ant-Miner was significantly more accurate that J48 in two (out of five) data sets, with the reverse being true in just one data set.

There are several possible avenues for future research. It would be interesting to investigate different rule evaluation measures in order to optimise the quality

of constructed rules. Different variations of the rule pruning procedure could prove to be more effective. Producing an unordered rule set instead of an ordered rule list could give more flexibility to the algorithm. Finally, it would be interesting to apply *h*Ant-Miner to more bioinformatics data sets.

Acknowledgements. The authors acknowledge the financial support from an European Union's INTERREG project (Ref. No. 162/025/361). Fernando Otero also acknowledges further financial support from the Computing Laboratory, University of Kent.

References

1. The Gene Ontology Consortium: Gene Ontology: tool for the unification of biology. Nature Genetics 25, 25–29 (2000)
2. Freitas, A., de Carvalho, A.: A tutorial on hierarchical classification with applications in bioinformatics. In: Taniar, D. (ed.) Research and Trends in Data Mining Technologies and Applications, pp. 175–208. Idea Group, USA (2007)
3. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
4. Jensen, L., Gupta, R., Stærfeldt, H., Brunak, S.: Prediction of human protein function according to Gene Ontology categories. Bioinformatics 19(5), 635–642 (2003)
5. Lægreid, A., Hvidsten, T., Midelfart, H., Komorowski, J., Sandvik, A.: Predicting gene ontology biological process from temporal gene expression patterns. Genome Research 13(5), 965–979 (2003)
6. Bi, R., Zhou, Y., Lu, F., Wang, W.: Predicting Gene Ontology functions based on support vector machines and statistical significance estimation. Neurocomputing 70, 718–725 (2007)
7. Blockeel, H., Schietgat, L., Struyf, J., Džeroski, S., Clare, A.: Decision Trees for Hierarchical Multilabel Classification: A Case Study in Functional Genomics. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 18–29. Springer, Heidelberg (2006)
8. Japkowicz, N., Stephen, S.: The class imbalance problem: a systematic study. Intelligent Data Analysis 6, 429–450 (2002)
9. Barutcuoglu, Z., Schapire, R.E., Troyanskaya, O.G.: Hierarchical multi-label prediction of gene function. Bioinformatics 22(7), 830–836 (2006)
10. Kiritchenko, S., Matwin, S., Famili, A.F.: Functional annotation of genes using hierarchical text categorization. In: BioLINK SIG: Linking Literature, Information and Knowledge for Biology (2005)
11. Clare, A., Karwath, A., Ougham, H., King, R.: Functional bioinformatics for *Arabidopsis thailana*. Bioinformatics 22(9), 1130–1136 (2006)
12. Parpinelli, R., Lopes, H., Freitas, A.: Data mining with an ant colony optimization algorithm. IEEE Transactions on Evolutionary Computation 6(4), 321–332 (2002)
13. Otero, F., Freitas, A., Johnson, C.: *c*Ant-Miner: an ant colony classification algorithm to cope with continuous attributes. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 48–59. Springer, Heidelberg (2008)
14. Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., Walter, P.: The Molecular Biology of the Cell, 4th edn. Garland Press (2002)
15. Witten, H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
16. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)

Conquering the Needle-in-a-Haystack: How Correlated Input Variables Beneficially Alter the Fitness Landscape for Neural Networks

Stephen D. Turner, Marylyn D. Ritchie, and William S. Bush

Center for Human Genetics Research, Department of Molecular Physiology & Biophysics, Vanderbilt University, Nashville, TN, USA
{stephen,ritchie,bush}@chgr.mc.vanderbilt.edu

Abstract. Evolutionary algorithms such as genetic programming and grammatical evolution have been used for simultaneously optimizing network architecture, variable selection, and weights for artificial neural networks. Using an evolutionary algorithm to perform variable selection while searching for non-linear interactions is akin to searching for a needle in a haystack. There is, however, a considerable amount of correlation among variables in biological datasets, such as in microarray or genetic studies. Using the XOR problem, we show that correlation between non-functional and functional variables alters the variable selection fitness landscape by broadening the fitness peak over a wider range of potential input variables. Furthermore, when sub-optimal weights are used, local optima in the variable selection fitness landscape appear centered on each of the two functional variables. These attributes of the fitness landscape may supply building blocks for evolutionary search procedures, and may provide a rationale for conducting a local search for variable selection.

Keywords: Neural networks, machine learning, gene-gene interaction, correlation, linkage disequilibrium, fitness landscape.

1 Introduction

1.1 Artificial Neural Networks

Artificial neural networks (ANNs) are a robust and flexible modeling technique that attempt to mimic the basic structure and function of biological neurons to solve complex problems. ANNs have been applied to a plethora of research fields, including robotics, speech recognition, optical character recognition, task scheduling, and industrial processing among many others. ANNs have also been widely applied to various problems in biological science, including microarray data analysis [1], human linkage analysis [2;3], genetic association studies [4], medical expert systems [5-7], survival analysis [8], and protein folding [9].

The traditional approach for applying ANNs to a classification problem is to specify a network architecture, select which variables are included as inputs to the network, and fit network weights using a gradient-descent based approach such as

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 80–91, 2009.

backpropagation [10]. When applied to biological data sets, this approach is problematic for several reasons. First, the proper network architecture is often unknown and must be specified arbitrarily by the user. Secondly, a collection of input variables must be selected. In many engineering applications, this set of variables is small, but in biological data, many thousands of variables are collected and it is impractical to include all variables as input for the ANN. Therefore, only a subset of the collected variables are included as input. Thirdly, the weight space for fitting noisy biological data with inconsistent signals may contain local optima that will trap gradient-descent weight optimization algorithms.

Recently, numerous evolutionary search strategies have been applied to ANN classification problems to reduce the issues associated with the traditional ANN approach[11]. Genetic Programming Neural Networks [12;13] and Grammatical Evolution Neural Networks [14] use genetic programming [15] or grammatical evolution [16] to evolve populations of neural networks for human genetics classification problems. These populations are a heterogeneous mix of architectures, weights, and input variables which undergo mating, crossover, and recombination to ultimately identify an optimum ANN solution. These approaches use a single search strategy to explore a broad composite space of architectures, weights, and input variables. Particle swarm optimization [17] and simulated annealing [18] search algorithms have been applied to fitting neural network weights as an alternative to backpropagation. Others [19-21] have used a hybrid search strategy, applying an evolutionary computing technique to search input variable space and doing some degree of weight fitting using backpropagation or other techniques to explore the weight space.

Variable selection, however, still remains a major obstacle for evolutionary algorithms when there are a large number of possible input variables to choose from and only few are functionally related to the outcome of interest. In human genetic studies, for example, collecting information on over 1 million genetic variants per individual is becoming a common practice. Furthermore, it is widely accepted that gene-environment interactions and epistasis, or gene-gene interactions, are ubiquitous given the complex biomolecular interactions that are essential for regulation of gene expression and complex metabolic networks [22], and are likely to play a role in influencing human disease [23]. As such, the complex diseases being studied are most likely driven by an intricate architecture of multiple genetic and environmental variables. The number of possible combinations of 1 million genetic variants is enormous, yet only a very small fraction of these may be truly involved in influencing a disease outcome. Finding genetic variants or interactions between them that contribute to disease risk determination has been likened to a needle in a haystack problem [24]. Since it is a common practice in human genetics to only test for interactions among variables with significant main effects, the difficulty of the problem is especially worse when individual variants have no detectable effect alone [25]. Such would be the case in an example problem where disease risk is increased for an individual who has a particular form of a variant at one of two genetic loci, but not both [26], representing an exclusive-OR (XOR) function of the two genetic variants. In this case, it would be difficult for an evolutionary algorithm such as GP or GE to select the functional variables as inputs to an ANN because the variable selection fitness landscape would not have contours upon which the algorithm could learn.

1.2 Correlation among Input Variables Due to Linkage Disequilibrium

The properties of evolutionary optimization and neural networks have largely been evaluated in an engineering context, typically for well characterized problem-solving tasks with many available training samples. Biological data sets, in particular those that contain human biomarker readings, are expensive to collect and generally have comparatively few samples. One other common property of biological marker sets is that they likely contain some degree of correlation between the markers. Microarray data, for example, captures the transcriptional activity of a collection of genes, and because multiple genes bind the same sets of regulatory co-factors, there is often a high degree of co-regulation among gene transcripts [27]. Genetic marker sets contain patterns of linkage disequilibrium formed as groups of nearby markers are transmitted together through multiple generations of human sub-populations [28,29]. Figure 1 is a plot showing linkage disequilibrium among pairs of genetic markers in a 50 kilobase region on human chromosome 6 in a Caucasian population.

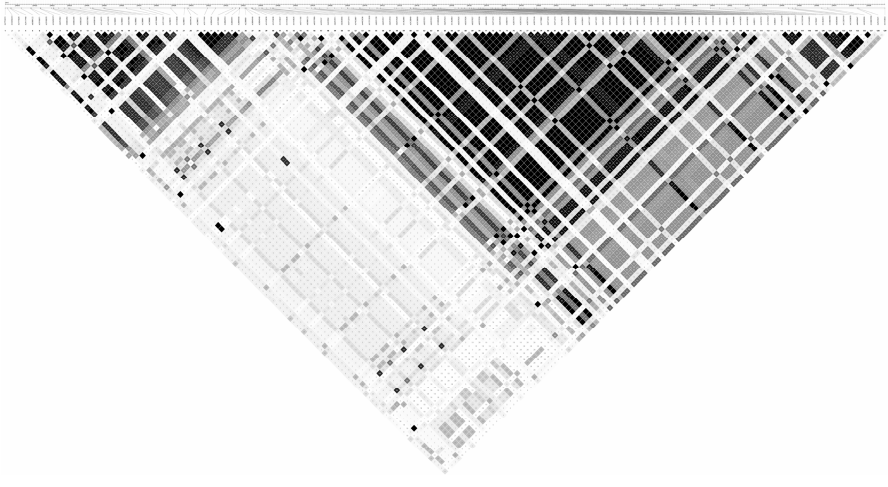


Fig. 1. Linkage disequilibrium among pairs of genetic markers in a 50 kilobase region on human chromosome 6 in a Caucasian population. Black indicates perfect correlation (r^2=1), white indicates no correlation (r^2=0) and shades of gray between white and black represent increasing correlation between genetic variants.

It is this correlation, in fact, that makes population-based disease gene mapping possible, since most studies rely on an indirect association of a neutral variant (which is correlated to an influential variant) to a disease state [30]. When all variables in the dataset are independent, using an evolutionary algorithm to search for interactions among genetic variants that contribute to disease risk is likely a needle in a haystack problem. However, we hypothesize that correlation between nonfunctional input variables to the functional variables will spread the signal over a wider range of inputs, making the problem more suitable for an evolutionary approach to variable selection. We will address this question by examining the surface of the variable selection fitness landscape when there are multiple variables with varying degrees of correlation to the functional variables.

2 Methods

Data simulation was implemented using the Perl scripting language. Two functional input variables were drawn randomly from a binomial distribution with p=0.5 for 1000 samples. The outcome variable y is the logical XOR function of the two relevant variables x_1 and x_2. 98 other nonfunctional variables were simulated with varying degrees of correlation to the functional variables x_1 and x_2. Correlation between the functional variables and other variables in the dataset is given by the square of the correlation coefficient [31]:

$$r_{ij}^2 = \frac{(n_{ij} - m_{ij})^2}{m_{ij}(1 - p_{i+})(1 - p_{+j})}, \tag{1}$$

where n_{ij} is the number of training examples with a particular combination of values for each variable, m_{ij} is the expected number of training examples if there were no correlation, and p is the proportion of individuals with a value of "1" for the ith or jth factor (0.5 in these experiments). When the frequency of x_i=1 is 0.5, solving for p_{ij} equation (1) simplifies to:

$$p_{ij} = 0.25\sqrt{r^2} + 0.25 \cdot \tag{2}$$

This indicates the joint probability of any other variable x_i=1 when the functional variable to which it is correlated is also equal to 1. Therefore, the conditional probability is given by:

$$p_{(x_i=1|x_1=1)} = \frac{\left(0.25\sqrt{r^2} + 0.25\right)}{0.5} = 0.5\sqrt{r^2} + 0.5 \cdot \tag{3}$$

We define c*orrelation magnitude* as the maximum correlation between each of the functional variables to two other nonfunctional variables in the dataset. In the results presented in Figure 3, this correlation extends to the 20 surrounding variable pairs decreasing exponentially to zero.

We implemented the 2-2-1 ANN shown in Figure 2 with sigmoid activation functions using Perl. The output y of each node in the network is given by:

$$y = f_s\left(\sum_{i=0}^{n} w_i x_i + w_{0i}\right). \tag{4}$$

where $\mathbf{w} = (w_1, w_2, ..., w_n) \in \mathbf{R}$ is a vector of weights applied to the input variables x_i, and where w_0 is the bias weight to each node, and f_s is the sigmoid activation function given by:

$$f_s(a) = \frac{1}{1 + e^{-a}} \cdot \tag{5}$$

The network in figure 2 was initialized with the weights shown to be capable of fitting an XOR function, adapted from [32], as shown in Figure 2 and in Table 1. Mean square error was calculated as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_p - y_t\right)^2 , \tag{6}$$

where y_p is the outcome as predicted by the network, and y_t is the target value of the XOR function of the two functional variables, and n is the total number of individuals. Fitness shown in Figure 3 was calculated as 1-MSE.

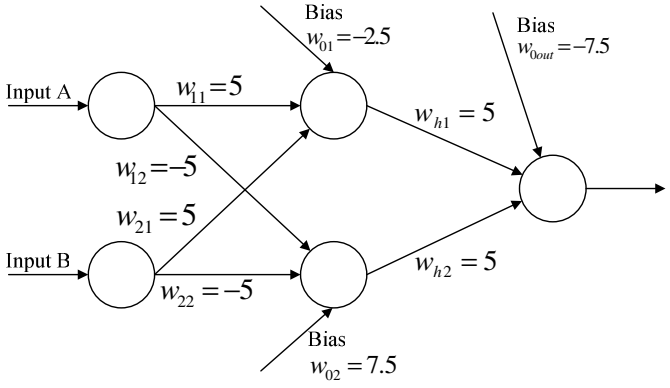


Fig. 2. 2-2-1 neural network to solve the XOR problem

Figure 1 was produced in Haploview [33] using HapMap data [29]. Figure 3 was created using the Fields package [34] in R [35]. All Perl and R code used in these experiments as well as a high resolution color version of Figure 3 are available at http://chgr.mc.vanderbilt.edu/ritchielab/projects/XOR/fitness_landscape.zip.

3 Results

We first examined the variable selection fitness landscape for all possible combinations of 100 variables when all variables in the dataset were independent. These results are illustrated in the top left panel of Figure 3. The plot represents the surface of the fitness (1-MSE) landscape as a function of the combination of predictor variables included as inputs to the 2-2-1 neural network depicted in Figure 2. The two functional variables are indicated by arrows in the top panels. When there is no correlation present, the correct solution that includes the two functional variables is the only model with high fitness, and appears as a spike in the variable selection fitness landscape surface (top panel in column A). As the correlation magnitude increases, the fitness of solutions in the variable selection fitness landscape transforms from a sharp spike into a cone-shaped hill that stretches over a larger part of the landscape (column A).

We also investigated the variable selection fitness landscape with increasing correlation magnitude when weights were not optimized to solve the XOR problem. Each optimized weight in the network was increased or decreased by a value randomly

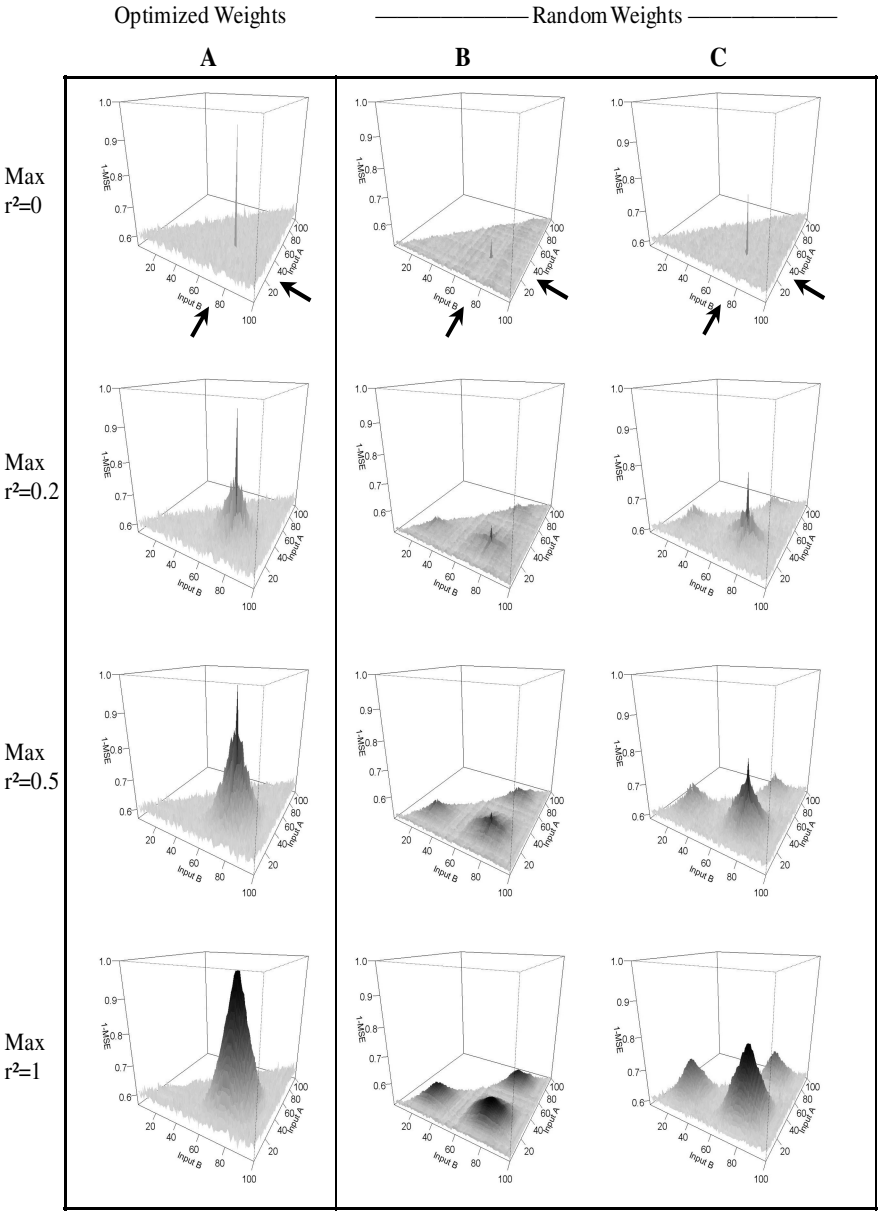


Fig. 3. Variable selection fitness landscapes with fitted weights (left column) and random weights (right 3 columns) with increasing levels of correlation magnitude

drawn between 0 and 5. 20 different sets of random weight combinations were evaluated, and we observed a common pattern in the variable selection fitness landscape. Two different representative sets of landscapes are shown in columns B and C of

Table 1. Values for optimized and random weights in networks used in Figure 3.

Column	w_{11}	w_{12}	w_{21}	w_{22}	w_{h1}	w_{h2}	w_{01}	w_{02}	w_{0out}
A (optimal)	5	-5	5	-5	5	5	-2.5	7.5	-7.5
B (random)	5	-3.1	6.3	-4	5.2	5.8	-1.5	7.9	-5.6
C (random)	7	-4.5	5.6	-4.9	5.5	6.1	-2.2	9	-6.9

Figure 3. These plots represent the variable selection fitness landscape of a 2-2-1 network with weights not optimal for the XOR problem. Values for the XOR-optimized and randomly perturbed weights used to generate Figure 3 are shown in Table 1. All 20 random weight combinations and their variable selection fitness landscapes are available at:
http://chgr.mc.vanderbilt.edu/ritchielab/projects/XOR/fitness_landscape.zip.

When all 100 variables are independent, a gain in fitness is seen only when both functional variables are inputs to the network, regardless of the weights (Figure 3, top left panel). Having non-optimal weights (Figure 3, top row) reduces the network fitness, but the functional combination is still the only peak in the fitness landscape.

As correlation magnitude increases, non-optimal weights in the network produce local optima in the variable selection fitness landscape. These optima arise at points where both input variables are correlated to the same single functional variable (autocorrelated). In some cases, these optima, due to autocorrelation, have similar fitness to the combination of inputs containing both functional variables. For random weight combination B when $r^2=1$ (Figure 3, column B, bottom row), the fitness of the global optima and local optima differ by only 0.0133. It is also notable that the local optima of some random weight combinations have higher fitness than the global optimum for other weight combinations. For example, with random weight combination C when $r^2=1$ (Figure 3, column C, bottom row), the MSE of the local optima is 0.72, which is higher than MSE of the global optimum (0.60) in combination B when $r^2=1$ (Figure 3, column B, bottom row). For all other weight combinations, the amplitude of this effect was different but the general shape and contour of the landscape was fundamentally similar. These results clearly show that correlation among variables in the dataset has a significant impact on the fitness landscape of the XOR problem.

4 Discussion

We investigated the variable selection fitness landscape for a neural network solution to the XOR problem with and without correlation of other variables to functional variables that determine the outcome. We have shown that when all variables in the dataset are independent, the network that contains both functional variables has very high fitness, representing a spike in the surface of the fitness landscape. All other combinations of variables have low fitness. We have also shown that as correlation magnitude increases, the fitness of networks containing those correlated variables also increases proportionally. The resulting fitness landscape transforms from a sharp spike to a broad hill as correlation magnitude increases.

These results have immediate implications for methods that employ evolutionary algorithms for variable selection. In fields where many attributes may be collected but only a few are truly associated to the outcome, correlation among the variables can potentially aid the search. The outcome in our simulations involved an interaction between 2 variables in a total set of only 100 variables. In human genetics studies, for example, it is very common to have data on over 1 million genetic markers. If all of these markers were independent, using an evolutionary algorithm to search for interactive effects would truly be a search for a needle in haystack, and would require excessive training before an acceptable solution is attained. In biological data however, correlation inevitably exists between variables. In human genetics, correlation is observed due to linkage disequilibrium, because sets of polymorphic variants that are physically linked to each other are likely inherited together through generations in a population. This correlation is very strong among closely linked markers, and weak correlation can extend long range to hundreds or thousands of other variants. Furthermore, if a genetic variant is under positive selection (e.g. a protective variant), very strong linkage disequilibrium will extend long range to many other neutral variants surrounding it [36-38]. This phenomenon has been observed in several human genes, including monoamine oxidase A [39], the multidrug resistance gene *MDR1* [40], and the dopamine receptor D4 [41]. In cases such as these, when neutral variables are correlated to functional variables, the fitness landscape is likely to contain a broad peak much more amenable to an evolutionary approach to variable selection.

Furthermore, another recent study provided some empirical evidence that linkage disequilibrium improves the sensitivity of grammatical evolution neural networks to detect a purely interactive effect embedded in a larger pool of non-associated variables [42]. In this procedure, the input variables, architecture, and weights of the NN are all optimized simultaneously by the grammatical evolution process. Our observations may explain the benefit that linkage disequilibrium provides to the GE search procedure.

In our examination of the variable selection space, increasing amounts of correlation to the functional variables produced local optima when the optimal NN weights are randomly perturbed. These optima correspond to points of autocorrelation of each of the two relevant variables in the XOR function. While local optima are generally considered detrimental to a search procedure, in this circumstance, the elevated fitness of NNs containing autocorrelated variables may provide building blocks to the grammatical evolution procedure.

When there is no correlation in the variable selection space, an evolutionary search procedure must include both functional variables in the network in order to see an increase in fitness, regardless of NN weights. With increasing amounts of correlation in the variable space, local optima exist in the joint variable-weight space that center around each of the two functional variables. Effectively, the evolutionary procedure can explore the weight space to find a NN function that provides elevated fitness to variables correlated to *one* of the two XOR functional variables. This would allow NNs containing only one of the two functional variables to be propagated in the solution population where it could eventually be crossed with a NN containing the other functional variable. As such, correlation in the variable selection space provides building blocks for the XOR function.

With a needle in the haystack problem like the XOR, the likelihood of finding the global optimum depends on the ability of the search algorithm to pair the two functional variables together in the NN solution. Correlation can induce the formation of local optima that center around one of the functional variables under some weight combinations. Therefore, correlation provides building blocks to evolutionary procedures that explore the joint variable-weight space. Although not explicitly tested here, it is likely that this phenomenon is not limited to the XOR function alone, and may extend to other complex non-linear problems.

Finally, these results may lead to innovations in searching the variable selection space when correlation is present. In genetics studies, correlation is highest among genetic variants in close physical proximity due to linkage disequilibrium. This creates a correlation structure that is tied to spatial relationships between variables.

Current hybrid search strategies use a combination of evolutionary algorithms to select input variables and architectures, and local optimization routines such as back-propagation to fit NN weights. Our observations in this study suggest that when there is correlation between variables, contours appear in the variable selection space which a local optimization routine could exploit, even when NN weights are non-optimal. When a spatial relationship exists, this local search could be accomplished by exploring neighboring variables using a fixed set of weights. When there is no natural spatial relationship between variables, variable position can be artificially defined based on correlation structures within the data. Search routines that supplement evolutionary algorithms with local optimization in both weight fitting *and* variable selection may have an advantage when correlation exists.

This study is a qualitative examination of the variable selection fitness landscape for the XOR problem. Future work should include quantitative examination of locality and problem hardness for this as well as other nonlinear problems in addition to the XOR. Furthermore, this study examines datasets with very uniform patterns of correlation. As seen in Figure 1, realistic correlation structures in biological data are much less uniform, and include long range correlation as well as gaps in strong, short range correlation. Finally, fitness landscapes we show in figure 3 are three-dimensional slices of a much larger, higher dimensional landscape that cannot easily be visualized. However, the local optima that appear among correlated variables centered around one of the two functional variables as shown in figure 3 are still present even in higher dimensional model space. More statistically quantitative power studies should be performed to actually test different evolutionary algorithms on problems such as these with varying degrees and distribution of correlation among the independent variables. This should reveal how different correlation structures relate to increases in sensitivity/specificity or decreases of training time, and will help determine which evolutionary search strategies are optimal for biological applications with characteristic data correlation structures.

Acknowledgements

This work was supported by NIH grants 5T32GM080178-02 and HL65962.

References

1. Linder, R., Richards, T., Wagner, M.: Microarray data classified by artificial neural networks. Methods Mol. Biol. 382, 345–372 (2007)
2. Lucek, P., Hanke, J., Reich, J., Solla, S.A., Ott, J.: Multi-locus nonparametric linkage analysis of complex trait loci with neural networks. Hum. Hered. 48(5), 275–284 (1998)
3. Marinov, M., Weeks, D.: The complexity of linkage analysis with neural networks. Human Heredity 51, 169–176 (2001)
4. Ott, J.: Neural networks and disease association studies. American Journal of Medical Genetics (Neuropsychiatric Genetics) 105(60), 61 (2001)
5. Lisboa, P.J., Taktak, A.F.: The use of artificial neural networks in decision support in cancer: a systematic review. Neural Netw. 19(4), 408–415 (2006)
6. Ohlsson, M.: WeAidU-a decision support system for myocardial perfusion images using artificial neural networks. Artif. Intell. Med. 30(1), 49–60 (2004)
7. Porter, C.R., Crawford, E.D.: Combining artificial neural networks and transrectal ultrasound in the diagnosis of prostate cancer. Oncology (Williston. Park) 17(10), 1395–1399 (2003)
8. Sato, F., Shimada, Y., Selaru, F.M., Shibata, D., Maeda, M., Watanabe, G., Mori, Y., Stass, S.A., Imamura, M., Meltzer, S.J.: Prediction of survival in patients with esophageal carcinoma using artificial neural networks. Cancer 103(8), 1596–1605 (2005)
9. Meiler, J., Baker, D.: Coupled prediction of protein secondary and tertiary structure. Proc. Natl. Acad. Sci. U.S.A 100(21), 12105–12110 (2003)
10. Bishop, C.M.: Neural Networks for Pattern Recognition, pp. 1–482. Oxford University Press, London (1995)
11. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 87(9), 1423–1447 (1999)
12. Motsinger, A.A., Lee, S.L., Mellick, G., Ritchie, M.D.: GPNN: power studies and applications of a neural network method for detecting gene-gene interactions in studies of human disease. BMC Bioinformatics 7, 39 (2006)
13. Ritchie, M.D., Coffey, C.S.M.J.H.: Genetic programming neural networks: A bioinformatics tool for human genetics. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 438–448. Springer, Heidelberg (2004)
14. Motsinger-Reif, A.A., Fanelli, T.J., Davis, A.C., Ritchie, M.D.: Power of grammatical evolution neural networks to detect gene-gene interactions in the presence of error. BMC. Res. Notes 1, 65 (2008)
15. Koza, J., Rice, J.: Genetic generation of both the weights and architecture for a neural network. IEEE Transactions II (1991)
16. O'Neil, M., Ryan, C.: Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language, 1st edn. Kluwer Academic Publishers, Norwell (2003)
17. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE International Conference on Neural Networks Proceedings, vol. 4, pp. 1942–1948 (1995)
18. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220, 671–680 (1983)
19. Skinner, A.J., Broughton, J.Q.: Neural networks in computational materials science: training algorithms. Modelling and Simulation in Materials Science and Engineering 3(3), 371–390 (1995)
20. Likartsis, A., Vlachavas, I., Tsoukalas, L.H.: A new hybrid neural-genetic methodology for improving learning. In: Ninth IEEE International Conference on Tools with Artificial Intelligence Proceedings, pp. 32–36 (1997)

21. Cantu-Paz, E., Kamath, C.: Evolving neural networks to identify bent-double galaxies in the FIRST survey. Neural Networks 16, 507–517 (2008)
22. Gibson, G.: Epistasis and pleiotropy as natural properties of transcriptional regulation. Theor. Popul. Biol., 49(1), 58–89 (1996)
23. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. Hum. Hered. 56(1-3), 73–82 (2003)
24. Weiss, K.M., Terwilliger, J.D.: How many diseases does it take to map a gene with SNPs? Nat. Genet. 26(2), 151–157 (2000)
25. Freitas, A.: Understand the Crucial Role of Attribute Interactions in Data Mining, 16th edn., pp. 177–199 (2001)
26. Li, W., Reich, J.: A complete enumeration and classification of two-locus disease models, 50th edn., pp. 334–349 (2000)
27. Komili, S., Silver, P.A.: Coupling and coordination in gene expression processes: a systems biology view. Nat. Rev. Genet. 9(1), 38–48 (2008)
28. International hapmap consortium; The International HapMap Project. Nature 426(6968), 789–796 (2003)
29. International hapmap consortium; A second generation human haplotype map of over 3.1 million SNPs. Nature 449(7164), 851–861 (2007)
30. Kruglyak, L.: Prospects for whole-genome linkage disequilibrium mapping of common disease genes. Nat. Genet. 22(2), 139–144 (1999)
31. Hill, W.G., Robertson, A.: Linkage disequilibrium in finite populations. Theoretical and Applied Genetics 38(6), 226–231 (1968)
32. Daqi, G., Yan, J.: Classification methodologies of multilayer perceptrons with sigmoid activation functions. Pattern Recognition 38(10), 1469–1482 (2005)
33. Barrett, J.C., Fry, B., Maller, J., Daly, M.J.: Haploview: analysis and visualization of LD and haplotype maps. Bioinformatics 21(2), 263–265 (2005)
34. Fields Development Team, Fields: Tools for Spatial Data, National Center for Atmospheric Research, Boulder, CO (2005), `http://www.cgd.ucar.edu/Software/Fields`
35. R Development Core Team, R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria (2005) ISBN 3900051070, `http://www.R-project.org`
36. Kimura, M.: The Neutral Theory of Molecular Evolution. Cambridge University Press, Cambridge (1985)
37. Sabeti, P.C., Reich, D.E., Higgins, J.M., Levine, H.Z., Richter, D.J., Schaffner, S.F., Gabriel, S.B., Platko, J.V., Patterson, N.J., McDonald, G.J., Ackerman, H.C., Campbell, S.J., Altshuler, D., Cooper, R., Kwiatkowski, D., Ward, R., Lander, E.S.: Detecting recent positive selection in the human genome from haplotype structure. Nature 419(6909), 832–837 (2002)
38. Smith, J.M., Haigh, J.: The hitch-hiking effect of a favourable gene. Genet. Res. 23(1), 23–35 (1974)
39. Gilad, Y., Rosenberg, S., Przeworski, M., Lancet, D., Skorecki, K.: Evidence for positive selection and population structure at the human MAO-A gene. Proc. Natl. Acad. Sci. U.S.A 99(2), 862–867 (2002)
40. Tang, K., Wong, L.P., Lee, E.J., Chong, S.S., Lee, C.G.: Genomic evidence for recent positive selection at the human MDR1 gene locus. Hum. Mol. Genet. 13(8), 783–797 (2004)

41. Ding, Y.C., Chi, H.C., Grady, D.L., Morishima, A., Kidd, J.R., Kidd, K.K., Flodman, P., Spence, M.A., Schuck, S., Swanson, J.M., Zhang, Y.P., Moyzis, R.K.: Evidence of positive selection acting at the human dopamine receptor D4 gene locus. Proc. Natl. Acad. Sci. U.S.A 99(1), 309–314 (2002)
42. Motsinger, A.A., Reif, D.M., Fanelli, T.J., Davis, A.C., Ritchie, M.D.: Linkage Disequilibrium in Genetic Association Studies Improves the Performance of Grammatical Evolution Neural Networks. In: IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology, pp. 1–8 (2007)

Optimal Use of Expert Knowledge in Ant Colony Optimization for the Analysis of Epistasis in Human Disease

Casey S. Greene, Jason M. Gilmore, Jeff Kiralis,
Peter C. Andrews, and Jason H. Moore

Dartmouth College, Lebanon, NH. USA
{Casey.S.Greene,Jason.M.Gilmore,Kiralis,Peter.C.Andrews,
Jason.H.Moore}@dartmouth.edu
http://www.epistasis.org

Abstract. The availability of chip-based technology has transformed human genetics and made routine the measurement of thousands of DNA sequence variations giving rise to an informatics challenge. This challenge is the identification of combinations of interacting DNA sequence variations predictive of common diseases. We have previously developed Multifactor Dimensionality Reduction (MDR), a method capable of detecting these interactions, but an exhaustive MDR analysis is exponential in time complexity and thus unsuitable for an interaction analysis of genome-wide datasets. Therefore we look to stochastic search approaches to find a suitable wrapper for the analysis of these data. We have previously shown that an ant colony optimization (ACO) framework can be successfully applied to human genetics when expert knowledge is included. We have integrated an ACO stochastic search wrapper into the open source MDR software package. In this wrapper we also introduce a scaling method based on an exponential distribution function with a single user-adjustable parameter. Here we obtain expert knowledge from Tuned ReliefF (TuRF), a method capable of detecting attribute interactions in the absence of main effects, and perform a power analysis at different parameter settings. We show that the expert knowledge distribution parameter, the retention factor, and the weighting of expert knowledge significantly affect the power of the method.

1 Introduction

Human geneticists are now capable of measuring over one million DNA sequence variations across thousands of individuals. Which of these sequence variations determine an individual's susceptibility to common human diseases such as cancer or cardiovascular disease? Making the problem more difficult, susceptibility to disease is likely determined not by a single gene but instead by the joint action of a number of interacting genes. Moore argues that these interactions, known as epistasis, are likely to be ubiquitous in common human diseases [1]. Moore's argument is based on four key pillars. These pillars are: the idea and concept of

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 92–103, 2009.

epistasis is grounded in almost 100 years of literature [2,3], interactions between biomolecules are ubiquitous in living systems, many single locus results from linkage and association studies have not replicated [4,5], and that when people search for epistasis using powerful methods they are frequently able to find examples of it [6,7]. Analysis methods which embrace the complexity present in biological systems are needed if we are to find predictors of common human diseases.

This complexity gives rise to a noisy environment and rugged fitness landscape. These data are frequently noisy as two individuals with the same values at the relevant attributes may have different disease states. This may be due to stochastic processes or environmental effects, e.g. someone unable to properly metabolize a fatty acid is likely to exhibit a disease state if they eat a diet high in that fatty acid but may not exhibit symptoms otherwise. The fitness landscape is rugged because models that contain some but not all of the relevant attributes may not have an accuracy greater than that of the surrounding noise.

Combining the difficulty of modeling nonlinear attribute interactions with the challenge of attribute selection yields, for this domain, what Goldberg calls a needle-in-a-haystack problem [8]. Under these models the learning algorithm is truly looking for a genetic needle in a genomic haystack. A recent report from the International HapMap Consortium suggests that approximately 300,000 carefully selected SNPs (i.e. single nucleotide polymorphisms which are frequently used measurements of genetic variation) may be necessary to capture all common variation across the Caucasian human genome [9]. Assuming this is true, though it is probably a lower bound, we would need to scan 4.5×10^{10} pairwise combinations of SNPs to find a genetic needle. The number of higher order combinations is astronomical. We look to biological systems for inspiration and methods which can solve these problems, not just because of the connection between method and application, but also because many of the problems natural systems cope with are share similarities with biological data.

Here we look to the system of ant colonies for inspiration. Ants search rugged and noisy landscapes for food and are able to rapidly take advantage of food sources when found. Our algorithm here should also balance this exploration and exploitation effectively. The idea to use algorithms inspired by ant colonies is not new. Indeed Dorigio, in 1991, discussed using ants as a metaphor for driving a positive feedback approach to search [10]. Furthermore we have shown that these ant systems can also work for this problem in human genetics when effective expert knowledge is provided to the algorithm [11]. Here we rigorously examine the presence and scaling of expert knowledge within these systems. We provide a thorough statistical analysis of the parameters using logistic regression, and suggest good parameters for use in genome-wide genetic analyses.

2 Ant Systems

Ant systems use a positive feedback approach to search which is modeled on the behavior of ants. This ant colony based approach is attractive for the area of human genetics because it is a straightforward population based approach which

is easily parallelizable and the method is conceptually simple. While the ant colony metaphor is most straightforward for spatial problems like the traveling salesman problem to which it was first applied [10], it has since been applied to areas from scheduling [12] to rule mining and discovery in biological data [13]. Dorigo and Stützle provide an introduction to ant systems and thorough review of the literature [14].

In an ant system, ants explore the search space and leave pheromone in quantities relative to the quality of their discoveries. Over time the pheromone evaporates. The quantity of this pheromone determines the chance that ants will explore similar regions in the future. With this addition and evaporation of pheromone, ant systems balance exploration of new areas and exploitation of areas that have previously contained good solutions. While there are new ant systems which exploit the behaviors of different types of ants [15], we utilize a traditional ant system here. We discovered in our work with genetic programming that expert knowledge is critical if machine learning algorithms are to succeed on this problem [16]. We have previously discovered with ant systems that an effective approach is to include this expert knowledge as heuristic information [11]. We use this heuristic information to change how pheromone is deposited across the search space. More detail on our specific ant system is provided in section 5.

3 Building Blocks from Tuned ReliefF (TuRF)

For our ant system we require a source of outside knowledge capable of preprocessing the input data and weighting attributes (SNPs) on how well they, in the context of other SNPs, are able to differentiate individuals with disease from those without. Kira and Rendell developed Relief which is capable of detecting attribute dependencies [17]. The approach Relief uses to detect interactions is conceptually simple. SNPs that distinguish genetically similar individuals in different classes are likely to be of interest. For any random individual, R_i, the nearest individual of the same class (H_i) and the nearest individual of the other class (M_i) are found. For each SNP, A, if R_i shares a value with H_i but not with M_i, the weight of A is increased. If R_i shares A with M_i but not with H_i, the weight of A is decreased. If R_i shares A with or differs from both H_i and M_i, the weight remains unchanged. This process of weighting attributes can be repeated for all individuals in the dataset. This neighbor-based process allows Relief to discover attribute dependencies when an interaction with no main effect exists. The algorithm produces weights for each attribute ranging from -1 (worst) to +1 (best). Kononenko improved upon Relief by choosing n nearest neighbors instead of just one [18]. This new algorithm, ReliefF, has been shown to be more robust to noisy attributes and missing data and is widely used in data mining applications [19]. Unfortunately the power of ReliefF is reduced in the presence of a large number of noisy attributes. This drove the development of Tuned ReliefF (TuRF) for situations where the number of noisy attributes is large. The TuRF algorithm systematically removes attributes that have low weights so that the weights of the remaining attributes can be re-estimated.

TuRF is significantly better than ReliefF in the domain of human genetics [20]. The ant system we examine uses TuRF as its source of expert knowledge.

4 Multifactor Dimensionality Reduction (MDR)

Here we develop an ACO framework to be available in version 2.0 of the Multifactor Dimensionality Reduction (MDR) software package. This package provides a user friendly cross-platform Java GUI appropriate for genome-wide genetic analysis. The MDR method has been developed as a nonparametric and model-free genetic data mining strategy for identifying combinations of SNPs that are predictive of a discrete clinical endpoint [21]. The MDR method has been successfully applied to detecting gene-gene interactions in a variety of common human diseases. At the core of the MDR approach is an attribute construction algorithm that creates a new attribute by pooling genotypes from multiple SNPs. Constructive induction using the MDR kernel is accomplished in the following way. Given a threshold T, a multilocus genotype combination is considered high-risk if the ratio of cases (subjects with disease) to controls (healthy subjects) exceeds or equals T, otherwise it is considered low-risk. Genotype combinations considered to be high-risk are labeled G1 while those considered low-risk are labeled G0. This process constructs a new one-dimensional attribute with levels G0 and G1. It is this new single variable that is returned as the quality measure in the ACO approach (Section 5). The MDR method is described in more detail by Moore et al. [22].

The MDR method is implemented in an open-source java software package which is freely available from www.epistasis.org. This MDR software package features a user-friendly GUI and is widely used for epistasis analysis in human genetics research (e.g. [23,24] among many others). This new version of MDR features an ant system based on the prototype described by Greene et al. [11]. Here we test this framework under a variety of parameter settings and assess the performance under different settings. We develop an exponential probability selection function. This method facilitates user controlled scaling of expert-knowledge weights to probabilities. The MDR software package also includes an implementation of TuRF which can calculate weights to be used as expert knowledge. What we learn here can be directly applied to an analysis of epistasis in human genetics using this MDR software package.

5 The MDR Ant Colony Optimization (ACO) Approach

In the ant system implemented in the MDR software, the goal is to select SNPs which effectively, in concert, determine an individual's risk of disease. We use TuRF weights (Section 3) as expert knowledge. This TuRF weighting algorithm is implemented within the software package. TuRF weights are distributed from -1 to 1 and must be transformed to selection probabilities for use in the algorithm. Using an exponential distribution function and a user-adjustable parameter, θ, the algorithm transforms TuRF weights to SNP selection probabilities. Specifically, order the attributes $A_1, A_2, \ldots, A_N$ so that their TuRF scores

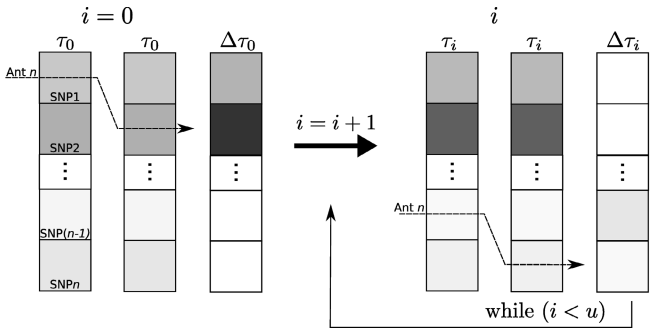


Fig. 1. In our ACO metaheuristic ants explore a pair of SNPs. From that pair of SNPs the amount of new pheromone, $\Delta\tau_i$, is calculated. $\Delta\tau_i$ is a combination of the quality of the pair, Q, and the expert knowledge score for that SNP, E. Shading here is representative of the strength of pheromone, τ, which begins as a probability from expert knowledge and changes based upon the attributes ants have chosen with each update, (i).

are increasing, and let $s_i - 1$ be the TuRF score of the i^{th} attribute, A_i. The s_i are the TuRF scores, but normalized so that they lie between 0 and 2. The probability that attribute A_i is selected, as given by the exponential function, is then

$$P(A_i \text{ is selected}) = \frac{1}{\sum_{k=1}^{N} \theta^{-s_k}} \theta^{-s_i},$$

where θ, the user-adjustable distribution parameter, satisfies $0 < \theta \leq 1$. Note that

$$P(A_i \text{ is selected}) = \theta^{s_{i+1}-s_i}\, P(A_{i+1} \text{ is selected}).$$

Thus if θ is near 1, attributes with higher scores are only slightly more likely to be selected than those with lower scores. Whereas if, for instance, $\theta = \frac{1}{2}$, attributes with higher scores are much more likely to be selected than those with lower scores. Furthermore, this function respects the intervals between TuRF scores. Larger intervals will produce larger differences between initialized SNP selection probabilities. These probabilities direct subsequent generations of ants by modifying the quantity of pheromone deposited. The weight of this information is determined by a parameter, β, in the pheromone update rule.

Equation 1 and Figure 1 show the pheromone update procedure. $\Delta\tau_{a,i}$ is the change in pheromone between updates.

$$\Delta\tau_{a,i} = \sum_{k=1}^{m} Q_{a,b} \times {E_a}^{\beta} \tag{1}$$

Here $Q_{a,b}$ is the MDR accuracy of a model containing that attribute a and the other attribute, b, chosen by ant k. E_a is the expert knowledge information

from TuRF for attribute a, and β is a parameter that determines the weighting of the expert knowledge, E. This update rule is used through u total updates.

6 Data Simulation and Analysis

Here we wish to explore the power of this stochastic search wrapper. Here power, expressed as a percentage, is the number of times out of 100 that the algorithm finds the relevant SNPs. To determine whether the algorithm is able to find these attributes, we simulate data where we know which SNPs determine disease risk. Here we generate thirty models of disease risk with each containing two relevant epistatic SNPs. These models span six broad sense heritabilities (0.025, 0.05, 0.1, 0.2, 0.3, and 0.4). For each heritability we generate five models. The heritability of a model determines how much an individual's genotypes at the functional SNPs affect that individual's disease status. The heritabilities we have chosen range from very low (0.025) to large (0.4). In all cases these models exhibit no main effects when the functional SNPs have minor allele frequencies of 0.4, the conditions we use for our dataset simulation. This means that the entire genetic effect will be due to the joint action of both SNPs and not due to information from a single SNP.

This provides a worst-case scenario for our machine learning algorithm, as it must search for the needle in the haystack. From these models we generate both 800 individuals with disease (cases) and 800 without disease (controls). We combine these into a single dataset and add an additional 998 noisy SNPs for a total of 1000 SNPs. We generate 100 of these datasets for each model. It is on these datasets where we know the relevant SNPs that we can effectively gauge the power of our method under various parameters.

Here we use results from the machine learning method TuRF (See section 3) as our source of expert knowledge. TuRF returns a weighting for each SNP which we can provide to our algorithm. We run TuRF using all 1600 individuals from the dataset. TuRF is a nearest neighbor approach, and we use the 200 nearest neighbors to provide weighting information. TuRF outperforms ReliefF in this domain by removing noisy attributes, i.e. those with the lowest weights, at each iteration. We choose to remove, at each iteration of the algorithm, the 50 (5%) attributes with the lowest weights before re-estimating the weights for the remaining attributes. We then provide the final TuRF weights to the ant system, which converts these weights into selection probabilities using our exponential distribution function.

Here we explore four major parameters of the MDR ant system. These are θ, β, a retention factor, and the number of ants and updates. The user-adjustable parameter for the exponential distribution function, θ, controls how expert knowledge is converted to probabilities (see Section 5). Here θ is varied from 0.9 to 1 in increments of 0.01 and, then from 0.99 to 1 in increments of 0.001. β determines the weight given to expert knowledge during the pheromone update procedure. Here we examine $\beta = 0$, 1, and 2. The retention factor determines how much weight information from previous iterations is given relative to information gained in the most recent iteration. Here we examine retention factors

of 0.1, 0.5, and 0.9. The number of ants and updates are controlled so that a maximum of 5000 total interactions can be explored. This is approximately equal to 1% of the total possible 2-way interactions in the dataset. We examine ant/update combinations of 10/500, 100/50, 250/20, and 500/10. In subsequent sections we refer to these combinations by the number of ants used. The number of updates is simply $5000/ants$.

To assess statistical significance we employ logistic regression. Logistic regression is attractive for these types of analyses because it allows for the statistically rigorous examination of how one or a number of continuous factors (e.g. parameters) influence a proportion (e.g. the number of correct answers for all total answers or power). In addition to allowing us to examine the significance of single parameters, logistic regression also allows us to determine whether interactions between parameters are statistically significant or simply due to chance. For more detailed information about logistic regression, the authors recommend Sokal and Rohlf [25] or Hastie, Tibshirani and Friedman [26]. Here we use the Design package [27] for the R statistical computing language [28] to perform logistic regression. We test all single parameter and pairwise interaction effects and assess significance. Here we use a singularity criterion of 1×10^{-20} and a maximum of 75 iterations for model fitting. We consider results significant when $p \leq 0.05$. This p-value can be thought of as the likelihood that chance alone explains the results. Therefore with this significance threshold of $p \leq 0.05$, we would only declare parameters significant one time out of twenty when there is no real effect.

7 Results

Not surprisingly, the performance of the algorithm varies greatly with the parameters chosen. Logistic regression shows that all single-parameter effects are significant at the ($p < 0.001$) level except for the number of ants and thus also the number of updates ($p > 0.05$). The expert-knowledge weighting factor, θ, had the greatest impact upon success. Figure 2 shows, for a single heritability, how the setting of θ affects power. Settings for θ from 0.99 to 1 are examined more closely because of the large change in power from 0.99 to 1.

The increased variability at higher values of θ is due to interactions between parameters (see Figure 4). Figure 3 shows that results are consistent across all six heritabilities. The great deal of consistent variability in these single-parameter results despite the large number of samples suggests that other parameters also play an important role in determining the likelihood of success of the algorithm and is indicative of interaction effects. Indeed logistic regression results indicate that β and the retention factor also have a statistically significant effect on power ($p < 0.0001$). Additionally, all two-way interactions between θ, β, and the retention factor are also statistically significant ($p \leq 0.05$).

In order to evaluate these interactions, we utilize interaction plots (Figure 4). These plots show how power varies with respect to a combination of two parameters. Here we examine interactions between θ and β and between β and the

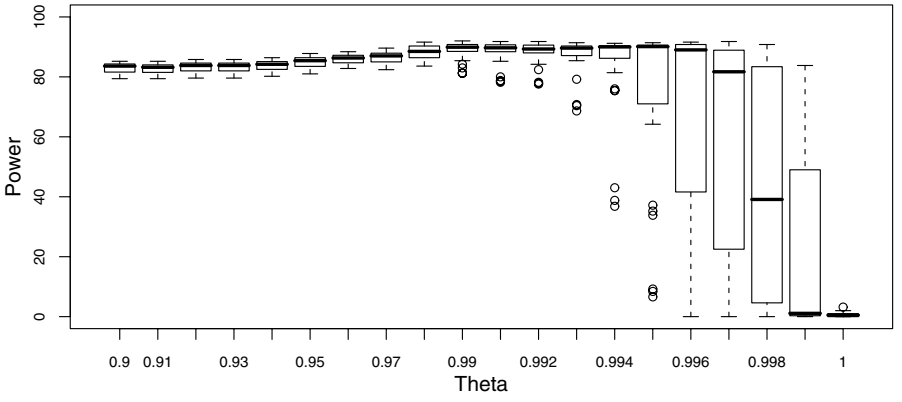


Fig. 2. θ shows the greatest single-parameter effect as measured by logistic regression. Here we show how power changes for a heritability of 0.1 as θ is varied from 0.9 to 1. Because the change in power between 0.99 and 1 is so great we have added additional points each 0.001 between those points. For visualization purposes the θ axis is not shown to scale. Each box extends between the 25th and 75th percentiles with a bold lines at the median (50th percentile). The whiskers of the plot extend through all non-outliers.

retention factor. We can see in the β and θ plot that with $\beta = 0$, power drops off rapidly beyond $\theta = 0.992$ but that with $\beta = 2$ power does not drop off rapidly until $\theta = 0.997$. Moreover we can see that while $\beta = 2$ seems to be the most reliable setting with regards to θ, it also leads to lower power than $\beta = 1$ when retention is 0.5 or 0.9.

Based on these results we suggest parameter settings of $\theta = 0.99$, $\beta = 1$, and a retention factor of 0.9. These parameters use a moderate weight for expert knowledge, retain much of the information learned by prior generations, and balance exploration and exploitation of expert knowledge. This value of θ provides a useful setting for the distribution parameter when TuRF is used as an expert knowledge source. While the combination of ants and updates is not a significant predictor, using 500 ants for 10 updates performs well and should be an appropriate setting. Greater numbers of ants and updates should be used when feasible given computational constraints.

8 Discussion

These results make clear that expert knowledge will be a crucial component if studies aimed at analyzing large scale association studies for interactions are to succeed. Furthermore the importance of developing an understanding of the expert knowledge source and the algorithm in conjunction are highlighted. Future studies should focus on using biological expert knowledge, in addition to statistical expert knowledge, in these search methods. We now possess a wealth of biological information about protein-protein interactions through databases

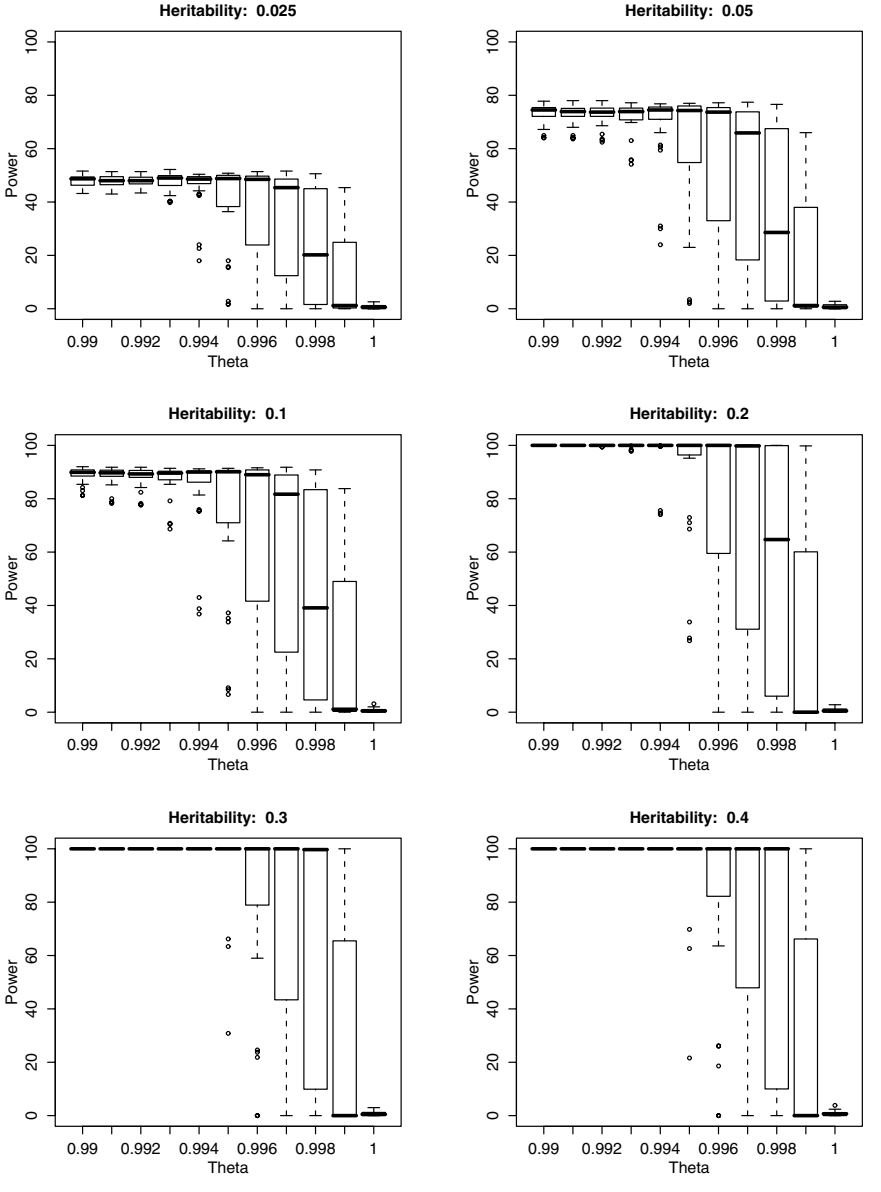


Fig. 3. Here that the change in power across heritabilities as θ changes from 0.99 to 1 is consistent. At different heritabilities there is still a great amount of variation at high values. Much of this variability comes from the effects of parameters other than theta on the power of the algorithm.

like String [29], the functions and processes genes are involved in with the Gene Ontology [30], and knowledge of biochemical pathways from KEGG [31]. When integrating these sources expert knowledge it will be critical to examine their role

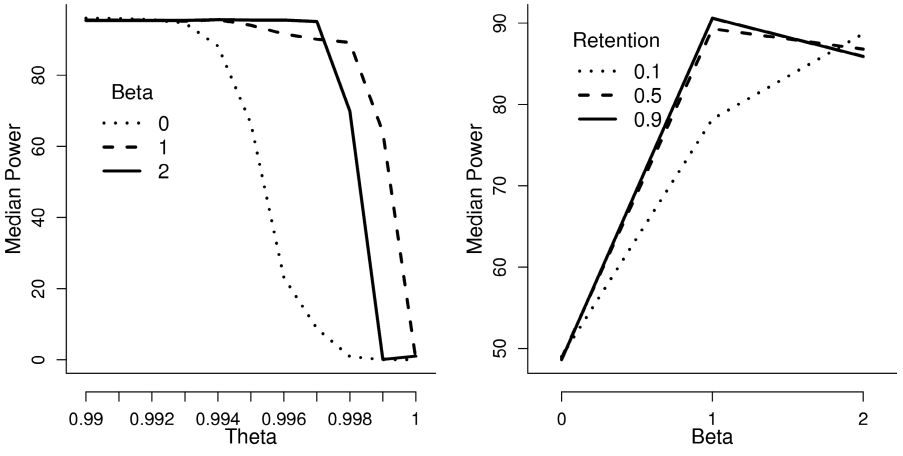


Fig. 4. In interaction plots parallel lines are indicative of additive parameter effects while non-parallel or intersecting lines are indicative of interaction effects. Here we see strong interaction effects. These are consistent with logistic regression which shows statistically significant pairwise interactions between these parameters.

with methods similar to those used here. We see numerous interactions between ant system parameters and the expert knowledge exponential distribution parameter. We therefore suggest logistic regression as a method of analyzing these types of results. Logistic regression gives the ability to answer detailed questions about parameters that lead to algorithm success and interactions between these parameters in a statistically defensible manner.

Further work in this area will require expert knowledge if these approaches are to discover predictors of common human diseases. Now the charge is to develop simple and efficient methods for attribute selection, to evaluate low parameter approaches for attribute selection and epistasis modeling, and to improve and develop sources of expert knowledge. Developing systems with few parameters and high power will make the application of these systems to real biomedical problems more routine. While the system here has four parameters, we are able to suggest $\theta = 0.99$, $\beta = 1$, and a retention factor of 0.9 as defaults when TuRF is used. The number of ants and updates should be maximized given computational constraints. Easy to use software packages and powerful methods will help put this class of methods in the modern geneticist's toolbox.

We can improve the expert knowledge component by improving our preprocessing methods or by using information derived from other biological data. Improvements to the power of ReliefF or TuRF will play a crucial role in our ability to find answers using nature inspired algorithms. Additionally, using biological expert knowledge from protein-protein interaction databases, the Gene Ontology, or known biochemical pathways we should be able to identify SNPs predictive of disease risk. Pattin et al. argue that once we successfully develop the methods to extract expert knowledge from protein-protein interaction databases,

we will improve our ability to identify important epistatic interactions in genome-wide studies [32]. By using biological knowledge to drive this search, the potential exists to enhance our comprehension of common human diseases. Eventually this should lead to an improvement in the prevention, diagnosis, and treatment of these diseases.

Ackowledgement

This work was supported by NIH grants LM009012, AI59694, and ES007373.

References

1. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. Human Heredity 56, 73–82 (2003)
2. Bateson, W.: Mendel's Principles of Heredity. Cambridge University Press, Cambridge (1909)
3. Shull, G.H.: Duplicate genes for capsule form in Bursa bursa-pastoris. J. Ind. Abst. Vererb 12, 97–149 (1914)
4. Hirschhorn, J.N., Lohmueller, K., Byrne, E., Hirschhorn, K.: A comprehensive review of genetic association studies. Genet. Med. 4, 45–61 (2002)
5. Finckh, U.: The future of genetic association studies in Alzheimer disease. Journal of Neural Transmission 110(3), 253–266 (2003)
6. Templeton, A.: Epistasis and complex traits. Epistasis and the Evolutionary Process, 41–57 (2000)
7. Leamy, L.J., Routman, E.J., Cheverud, J.M.: An Epistatic Genetic Basis for Fluctuating Asymmetry of Mandible Size in Mice. Evolution 56(3), 642–653 (2002)
8. Goldberg, D.E.: The Design of Innovation: Lessons from and for Competent Genetic Algorithms. Kluwer Academic Publishers, Norwell (2002)
9. The International HapMap Consortium: A haplotype map of the human genome. Nature 437(7063), 1299–1320 (2005)
10. Dorigo, M., Maniezzo, V., Colorni, A.: Positive feedback as a search strategy. Technical report 91-016, Dipartimento di Elettronica e Informatica, Politecnico di Milano (1991)
11. Greene, C.S., White, B.C., Moore, J.H.: Ant colony optimization for genome-wide genetic analysis. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 37–47. Springer, Heidelberg (2008)
12. Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M.: Ant system for job-shop scheduling. Belg. J. Oper. Res. 34, 39–53 (1994)
13. Parpinelli, R., Lopes, H., Freitas, A.: An Ant Colony Based System for Data Mining: Applications to Medical Data. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), pp. 791–797 (2001)
14. Dorigo, M., Stützle, T.: Ant Colony Optimization (2004)
15. Brutschy, A., Scheidler, A., Merkle, D., Middendorf, M.: Learning from house-hunting ants: Collective decision-making in organic computing systems. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 96–107. Springer, Heidelberg (2008)

16. Moore, J.H., White, B.C.: Genome-wide genetic analysis using genetic programming: The critical need for expert knowledge. In: Riolo, R., Soule, T., Worzel, B. (eds.) Genetic Programming Theory and Practice IV. Springer, Heidelberg (2007)
17. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: Machine Learning: Proceedings of the AAAI 1992 (1992)
18. Kononenko, I.: Estimating attributes: Analysis and extension of relief. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)
19. Robnik-Sikonja, M., Kononenko, I.: Theoretical and empirical analysis of relieff and rrelieff. Mach. Learn. 53(1-2), 23–69 (2003)
20. Moore, J.H., White, B.C.: Tuning relieff for genome-wide genetic analysis. In: Marchiori, E., Moore, J.H., Rajapakse, J.C. (eds.) EvoBIO 2007. LNCS, vol. 4447, pp. 166–175. Springer, Heidelberg (2007)
21. Ritchie, M.D., Hahn, L.W., Roodi, N., Bailey, L.R., Dupont, W.D., Parl, F.F., Moore, J.H.: Multifactor dimensionality reduction reveals high-order interactions among estrogen metabolism genes in sporadic breast cancer. American Journal of Human Genetics 69, 138–147 (2001)
22. Moore, J.H., Gilbert, J.C., Tsai, C.T., Chiang, F.T., Holden, T., Barney, N., White, B.C.: A flexible computational framework for detecting, characterizing, and interpreting statistical patterns of epistasis in genetic studies of human disease susceptibility. Journal of Theoretical Biology 241(2), 252–261 (2006)
23. Julià, A., Moore, J., Miquel, L., Alegre, C., Barceló, P., Ritchie, M., Marsal, S.: Identification of a two-loci epistatic interaction associated with susceptibility to rheumatoid arthritis through reverse engineering and multifactor dimensionality reduction. Genomics 90(1), 6–13 (2007)
24. Beretta, L., Cappiello, F., Moore, J.H., Barili, M., Greene, C.S., Scorza, R.: Ability of epistatic interactions of cytokine single-nucleotide polymorphisms to predict susceptibility to disease subsets in systemic sclerosis patients. Arthritis and Rheumatism 59(7), 974–983 (2008)
25. Sokal, R.R., Rohlf, F.J.: Biometry: the principles and practice of statistics in biological research, 3rd edn. W. H. Freeman and Co., New York (1995)
26. Hastie, T., Tibshirani, R., Friedman, J.: Elements of Statistical Learning, 1st edn. Springer, Canada (2001)
27. Harrell Jr., F.E.: Design: Design Package (2007)
28. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2008)
29. Snel, B., Lehmann, G., Bork, P., Huynen, M.A.: String: a web-server to retrieve and display the repeatedly occurring neighbourhood of a gene. Nucleic Acids Research 28(18), 3442–3444 (2000)
30. The Gene Ontology Consortium: Gene ontology: Tool for the unification of biology. Nature Genetics 25, 25–29 (2000)
31. Kanehisa, M., Goto, S.: Kegg: kyoto encyclopedia of genes and genomes. Nucleic Acids Research 28(1), 27–30 (2000)
32. Pattin, K., Moore, J.: Exploiting the proteome to improve the genome-wide genetic analysis of epistasis in common human diseases. Human Genetics 124(1), 19–29 (2008)

On the Efficiency of Local Search Methods for the Molecular Docking Problem

Jorge Tavares[1], Salma Mesmoudi[2], and El-Ghazali Talbi[2]

[1] Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139, USA
[2] INRIA Lille - Nord Europe Research Centre
40, Avenue Halley Bât.A, Park Plaza
59650 Villeneuve d'Ascq, France
jorge.tavares@ieee.org
{salma.mesmoudi,el-ghazali.talbi}@inria.fr

Abstract. Evolutionary approaches to molecular docking typically hybridize with local search methods, more specifically, the Solis-Wet method. However, some studies indicated that local search methods might not be very helpful in the context of molecular docking. An evolutionary algorithm with proper genetic operators can perform equally well or even outperform hybrid evolutionary approaches. We show that this is dependent on the type of local search method. We also propose an evolutionary algorithm which uses the L-BFGS method as local search. Results demonstrate that this hybrid evolutionary outperforms previous approaches and is better suited to serve as a basis for evolutionary docking methods.

1 Introduction

In general terms, molecular docking is a search problem that aims to find the best conformation and orientation of a molecule relative to the active site of a second target molecule with the lowest energy [1]. The typical case is to have a protein as a receptor, fixed in a three-dimensional coordinate system, and a ligand, which can be repositioned and rotated to dock it with the receptor. The docking problem is very difficult since the relative orientation and conformations of the two molecules must be considered. With both molecules flexible, usually the active site of the protein and the ligand, the problem becomes harder. In fact, a higher degree of flexibility implies a considerable increase of the search space size.

Evolutionary algorithms have recently become one of the dominant search techniques for docking methods [2] and proved to be very successful. Despite the fact that numerous applications of evolutionary algorithms exist, the number of studies which focus on *understanding why* the algorithm's components are successful is scarce. To the best of our knowledge, the first attempt made concerning this important topic can be found in [3]. Several parameters (e.g., population size) and some genetic operators are empirically investigated,

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 104–115, 2009.

as well as the efficiency of the use of a local search method. Some of our previous work reflects this concern of studying the effect of the components of an evolutionary algorithm on the search process [4,5]. However, in [3] it is concluded that local search methods might not be very helpful in the context of molecular docking. The several experiments provided there show that an evolutionary algorithm without local search but with proper genetic operators can perform equally well or even outperform the best evolutionary algorithms with local search for this problem, e.g., [6,2].

In this paper, we perform an empirical analysis on the evolutionary algorithm model usually adopted for molecular docking optimization [6,2]. We show, contrary to [3], that local search methods can be useful for this problem, when the appropriate method is selected, regardless of the genetic operators used. Results confirm that the popular Solis-Wets algorithm is not an ideal local search method for molecular docking since the solution's quality and the algorithm's efficiency is similar to evolutionary algorithms without auxiliary methods. In contrast, an evolutionary algorithm with the L-BFGS method, a powerful quasi-Newton conjugate gradient method, improves the results attained by an evolutionary algorithm tailored to molecular docking considerably.

The rest of the paper is structured as follows. Section 2 contains an overview of the evolutionary algorithm's components used in our experimentation. In section 3 we present the experiments and their results. Finally, section 4 contains the main conclusions.

2 Evolutionary Algorithms and Molecular Docking

Since 1993, evolutionary algorithms for the molecular docking problem can be found in the literature [7]. A comprehensive review of these efforts can be found in [8,2]. One of the most important works is the evolutionary algorithm proposed in [6], which is part of the package named *AutoDock*. The evolutionary algorithm is a conformational search method which uses an approximate physical model to evaluate possible protein-ligand conformations. It incorporates flexibility by allowing the ligand to change its conformation during the docking simulation. In addition, pairwise interactions between atoms are pre-calculated, which considerably speeds up the docking simulation. The approach uses an evolutionary algorithm with a local search method to search the space of possible protein-ligand conformations. When this method is applied, the genotype of the individuals is replaced with the new best solution found. This process is usually referred to as Lamarckian evolution. In our work, we adopt an experimental model which uses the main components from [6,3]. The main reason is that *AutoDock* serves as a basis for the large majority of evolutionary-inspired approaches (e.g., see [2]) and thus, the attained results here can be useful on a larger degree.

2.1 Encoding

During the docking process the protein remains rigid whilst the ligand is flexible. An individual represents only the ligand. The encoding is an indirect

representation. A genotype of a candidate solution is encoded by a vector of real-valued numbers which represent the ligand's translation, orientation and torsion angles [6]. Cartesian coordinates represent the translation which is defined by three variables in the vector. The orientation is determined by four variables in the vector, which define a quaternion. A quaternion can be considered to be a vector (x, y, z) which specifies an axis of rotation with an angle θ of rotation for this axis. For each flexible torsion angle one variable is used. The phenotype of a candidate solution is composed of the atomic coordinates that represent the three-dimensional structure of the ligand. The atomic structure is built from the translation and orientation coordinates in the ligand crystal structure with the application of the torsion angles.

2.2 Evaluation

An energy evaluation function is used to evaluate each individual. The fitness for each candidate solution is given by the sum of the intermolecular interaction energy between the ligand and the protein, and the intramolecular energy that arises from the ligand itself [6]. *AutoDock* uses an empirical free energy potential composed of five terms. The first three terms are pairwise interatomic potentials that account for weak long-range attractive forces and short-range electrostatic repulsive forces. The fourth term measures the unfavorable entropy of a ligand binding due to the restriction of conformational degrees of freedom. The fifth and last term uses a desolvation measure. Further details of the energy terms and how the potential is derived can be found in [6].

2.3 Genetic Operators

Common crossover and mutation operators are applied on the population. In *AutoDock* a standard two-point crossover is used. Cut points only occur between related genes, i.e., separating translational values, orientation values and rotation torsion angles into separate blocks. This is done to avoid disruption of useful parts of the solution [6]. However, for real-valued encodings it is recommended that operators designed to deal with this type of encoding are used. In our previous work [5], we analyze several common crossover operators, such as Simple Arithmetical crossover, Whole Arithmetical crossover, Discrete crossover, Simulated Binary Crossover and Blend-α crossover, in terms of their effect on representation properties, i.e., locality, heritability and heuristic bias. Results showed that Whole Arithmetical crossover is a good operator for molecular docking, which confirms some experimentations discussed in [3].

Since the encoding is a real-valued vector, mutation is performed by using operators based on evolutionary strategies. The genetic operator acts in the following way: when undergoing mutation, the new value for a gene x' is obtained from the old value x by adding a random real number sampled from a distribution $U(0, 1)$:

$$x' = x + \sigma \times N(0, 1) \tag{1}$$

where $N(0,1)$ is the standard Gaussian distribution. Here. an important aspect is determining the value for parameter σ. If it is set too low, exploitation overcomes exploration and if set too high *vice versa*. The value can be fixed or self-adapted (e.g., if an evolutionary strategy approach is used). In [3], the use of annealing schemes to control σ as a function of time, i.e., the number of generations, is proposed. The following scheme is used, scaled with 0.1:

$$\sigma(t) = \frac{1}{\sqrt{1+t}} \tag{2}$$

In the same manner as crossover, in [4,5] we also show that this kind of mutation operator is best suited for molecular docking, especially when comparing it with operators based on Cauchy distributions.

2.4 Local Search

In this work, we study the impact of two local search methods. The first technique is the algorithm used in [6]: the Solis-Wets method. The Solis-Wets algorithm is a direct search method with an adaptive step size, which performs a randomized local minimization of a given candidate solution. The process starts with a candidate solution $x \in \Re^n$ and for each step a deviate $\epsilon \in \Re$ is chosen from a normal distribution. In the case a better solution is found by adding or removing ϵ from x, the current solution is replaced with the new one. Depending on whether a new solution if found or not, a success or a failure is recorded. If several successes occur in a row, the variance of the normal distribution is adapted for the search to move more quickly. If the opposite occurs, the variance is adapted to focus the search. This is accomplished through a parameter, ρ. Moreover, a bias term is applied to drive the search in successful directions. The method terminates when a certain lower-bound threshold for ρ is passed or when a maximum number of steps is reached.

The second method used is the Broyden-Fletcher-Goldfarb-Shanno method (L-BFGS) [9]. L-BFGS is a powerful quasi-Newton conjugate gradient method, where both the function to minimize and its gradient must be supplied, but no a priori knowledge about the corresponding Hessian matrix is required. During local search, the maximum number of iterations that can be performed is specified by a parameter of the algorithm. However, the method stops as soon as it finds a local optimum, so the real number of iterations may be smaller than the specified value.

In a previous work concerning the analysis of local search methods for this problem [10], we analyzed the Solis-Wets algorithm and the simplex local search algorithm described by Nelder and Mead (NMS) for nonlinear, continuous function optimization [11]. The analysis was made in terms of locality and the attained results showed that both methods had a similar high impact on locality. As such, the performance of an evolutionary algorithm, as well as the solutions quality, is not considerably different when using one of these two methods. For this reason, the NMS method is not considered in this paper.

3 Experimentation and Analysis

Several instances from the *AutoDock* test suite are used to perform our optimization tests. The suite is composed of eight protein-ligand complexes. Each complex contains a macromolecule (the protein) and a small substrate or inhibitor molecule (the ligand). The structures of these molecular complexes have been obtained from the Protein Data Bank (PDB). Table 1 provides information about the complexes names as well as their PDB identification.

Table 1. X-ray crystal structures used in the docking experiments

Protein-ligand complex	PDB	Resolution	Torsion	Size
Alcohol dehydrogenase	1adb	2.4	14	21
Alpha-Thrombin	1bmm	2.6	12	19
Beta-Trypsin	3ptb	1.7	0	7
Carbonic anydrase	1nnb	2.3	9	16
Trypsin	1tnh	1.80	2	9
IGG1-KAPPA DB_3 FAB	2dbl	2.9	6	13
L-Arabinose-binding protein	7abp	1.67	4	11
HIV_{-1} Protease	1hvr	1.80	10	17

To evaluate a resulting ligand conformation we compare it with the experimental structures using the standard Cartesian root-mean-square deviation (RMSD):

$$RMSD_{lig} = \sqrt{\frac{\sum_{i=1}^{n} dx_i^2 + dy_i^2 + dz_i^2}{n}} \tag{3}$$

where n is the number of atoms in the comparison and dx_i^2, dy_i^2 and dz_i^2 are the deviations between the crystallographic structure and the corresponding coordinates from the predicted structure lig on Cartesian coordinate i. RMSD values below or near 2.0Å can be considered to be a success and the ligand is classified as being docked. On the other hand, a structure with a RMSD just less than 3.0Å is classified as partially docked. Thus, lower values mean that the observed and the predicted structures are similar.

We consider three versions of a standard evolutionary algorithm for our experimentation. The only difference between them is the local search method. The first variant does not include any type of local search, mimicking the algorithm presented in [3]. The second variant includes the Solis-Wets algorithm as local search, thus being equivalent to the algorithm contained in the *AutoDock* package [6]. The last evolutionary algorithm is our own proposal, which includes the L-BFGS method as local search. As for the rest of the evolutionary algorithm components, the basic algorithm for our experiments follows what was described in the previous sections in terms of representation, fitness function and genetic operators. The representation and evaluation is the one used by *AutoDock*. The

genetic operators used are the Whole Arithmetic crossover and the Gaussian-based mutation operator described in the previous section. Moreover, the evolutionary algorithm is a standard generational algorithm with stochastic tournament selection and weak elitism.

3.1 Settings

The parameter values were set heuristically, even though we did some additional tests and verified that, within a moderate variation, there was no significant difference in the outcomes. In any case, we did not perform a comprehensive study on the influence of different parameter settings and it is possible that a careful fine-tuning of some values could bring slight improvements to the achieved results. For all experiments, the settings of the tested algorithms are the following: Number of runs: 30; Population size: 150; Selection rate: 0.8; Crossover rate: 0.9; Mutation rate: 0.5; Tournament selection rate: 0.95; Tournament replacement rate: 0.9; Number of local search steps: 1000; Maximum number of fitness evaluations: 10 000 000. We should note that, although the maximum number of fitness evaluations is high, in the case where no improvement is found after 10 000 fitness evaluations, the algorithm stops. For both local search methods, a step counts as a single fitness evaluation.

3.2 Experiments

In table 2 we present an overview of the optimization results according to the energy fitness function. Each line displays the results for a complex, identified by the PDB label, with a given size. For each of three variants of the evolutionary algorithm, i.e., no local search (No LS), with the Solis-Wet algorithm (Solis-Wets) and the L-BFGS method (L-BFGS), a column is presented with three sub-columns. These contain the best energy value found during the 30 runs (Best), the average of the best energy values for 30 runs (Avg) and the corresponding standard deviation (Std). Bold values indicate the best values found.

A first observation of table 2 indicates a very important result: the evolutionary algorithm with the L-BFGS local search method achieves the best results for all instances. Moreover, it also attains the best results in terms of average and standard deviation. This clearly shows that the use of this local search method provides competitive results when compared to the alternative configurations. In fact, a closer look at the results reveals that the quality of the solutions found can be considerably. For the larger and more difficult complexes (1adb, 1bmm, and 1hvr) the order of magnitude can be the double, at least, when compared to the second best algorithm. For the large instance, 1adb, the L-BFGS attains an energy score of -18.30 whilst the variant without local search, the second best, only achieves -6.59. For the remaining complexes, especially the ones with a smaller size, the differences are not that striking. The complex 3ptb, which has the smaller size of the eight tested complexes, demonstrates this. The L-BFGS evolutionary algorithm has a best energy value of -6.33. The algorithm with the worst value is the no local search version with -5.49. In this case, there is a

Table 2. Summary of optimization results according to energy evaluation

Complex		No LS			Solis-Wets			L-BFGS		
Label	Size	Best	Avg	Std	Best	Avg	Std	Best	Avg	Std
1adb	21	-6.59	335.34	953.64	-3.85	192.36	559.51	**-18.30**	**-14.00**	**4.80**
1bmm	19	-3.94	5.00	11.68	-4.66	5.04	18.84	**-11.99**	**-5.52**	**2.73**
1hvr	17	-15.79	-8.67	9.39	-13.81	-10.13	4.07	**-32.43**	**-32.13**	**0.28**
1nnb	16	-4.53	-2.48	1.69	-5.18	-2.16	1.57	**-6.25**	**-4.63**	**1.50**
1tnh	9	-5.39	-2.73	1.36	-5.37	-2.49	1.14	**-6.06**	**-5.54**	**1.09**
2dbl	13	-10.03	-4.66	3.47	-11.43	-4.26	3.06	**-12.14**	**-10.04**	**2.99**
3ptb	7	-5.49	-3.66	1.02	-5.99	-4.16	1.18	**-6.33**	**-6.22**	**0.25**
7abp	11	-7.97	-6.78	0.90	-7.90	-6.84	1.14	**-9.12**	**-8.67**	**0.37**

difference of only 13.27% while for the larger complex (1adb), for the two same algorithms, the difference is 63.98%. This pattern is consistent when comparing the L-BFGS method to the other two. The larger the complexes, the larger the difference in results. The evolutionary algorithm with the L-BFGS clearly outperforms the other algorithms.

Another relevant aspect is the difference of performance between the two local search methods. As previously reported [3], an evolutionary algorithm without the Solis-Wets local search method can be efficient and competitive. Looking at table 2 we can agree with this conclusion since only in four instances (half the tested complexes) the Solis-Wets algorithm attains better results than without local search. Furthermore, the exact same behavior is exhibited in terms of the average of the best solutions found. This is an indication that these two variants of the evolutionary algorithm are more sensitive to the multi-modal landscape of the problem. The L-BFGS local search method enables the evolutionary algorithm to overcome the multiple local minima more easily.

In addition, the evolutionary algorithm with the L-BFGS method provides another indication that it is superior to the other versions. The column that displays the averages of the best solutions found during the 30 runs, shows for the L-BFGS method closer values to the best. Four instances have a proximity of values very small (0.92% for 1hvr, 1.7% for 3ptb, 4.99% for 7abp and 8.6% for 1tnh). The remaining ones have a small moderate distance (17.28% for 2dbl, 23.5% for 1adb and 25.88% for 1nnb) with the exception of 1bmm (distance of 53.97%). The proximity values between the best and the average are not reported for the other methods. The evolutionary algorithms without local search and with the Solis-Wets method show larger distances, the majority of them between 45% and 65% approximately. Although this indicates a more efficient local search method for the L-BFGS algorithm, it is important to point out that the method also reduces the population diversity substantially.

To establish if these differences are statistically significant, we performed the Wilcoxon rank sum test with significance value $\alpha = 0.01$. As expected, there are no significant differences between the evolutionary algorithms without local

Table 3. Summary of optimization results according to RMSD values

Complex		No LS			Solis-Wets			L-BFGS		
Label	Size	Best	Best-En	Avg	Best	Best-En	Avg	Best	Best-En	Avg
1adb	21	1.47	1.53	2.78	1.34	1.50	2.79	**0.30**	**0.51**	**1.64**
1bmm	19	2.06	2.06	4.18	1.29	1.29	3.89	**0.67**	**1.10**	**3.63**
1hvr	17	**0.29**	**0.29**	0.96	0.53	0.65	0.90	0.50	0.71	**0.61**
1nnb	16	0.44	0.76	**1.84**	0.62	**0.64**	2.76	**0.39**	0.74	**1.84**
1tnh	9	0.70	0.73	2.94	0.34	**0.34**	3.38	**0.20**	0.94	**1.16**
2dbl	13	0.57	0.78	2.42	0.53	0.53	2.64	**0.35**	**0.35**	**1.40**
3ptb	7	0.42	0.54	2.07	0.31	**0.31**	1.45	**0.24**	0.51	**0.93**
7abp	11	**0.22**	0.61	0.65	0.26	**0.45**	0.86	0.35	0.82	0.76

search and the Solis-Wets method. Significant differences are found between the evolutionary algorithm with the L-BFGS local search method and the other two algorithms.

It is important now to relate the energy results with the RMSD values. Table 3 displays the overview of these values. In the same way as before, each line displays the results for a complex. For each of three variants of the evolutionary algorithm, a column is presented with three sub-columns. These are: the best RMSD value found during the 30 runs (Best), the RMSD value correspondent to the best energy value found during the 30 runs (Best-En) and the average of the RMSD values for 30 runs (Avg). Bold indicates the best displayed values.

In terms of the RMSD values, the differences between the three approaches are not as distinctive as in the energy case. An overview of table 3 reveals that the L-BFGS method does not contain all the best values. Nevertheless, it still contains the majority of the best RMSD values. However, this table contains two important pieces of information. The first one indicates how well the algorithm is able to perfom in terms of RMSD, i.e., can the evolutionary algorithm attain low RMSD values? The answer is given by the Best and Avg columns. From the table, it is possible to conclude that all algorithms can discover solutions with low RMSD values. For example, 0.22 without local search for the 7abp complex, 0.34 with Solis-Wet method and 0.20 with L-BFGS method for the 1tnh complex. Moreover, the Best column reveals that the three evolutionary algorithms are able to reach consistently good values. With the exception of the evolutionary algorithm without local search for the 1hvr complex, every other value is lower than 1.5 and for most cases, even lower than 1.0. This performance is reinforced by the Avg column since the RMSD average of the best solutions found during the 30 runs is low. However, in this case the L-BFGS method attains a better performance compared to the remaining algorithms. The RMSD average on the eight tested instances is 1.5 for the L-BFGS method. The Solis-Wets method has an average of 2.33 and without local search we have a slight improvement: 2.23.

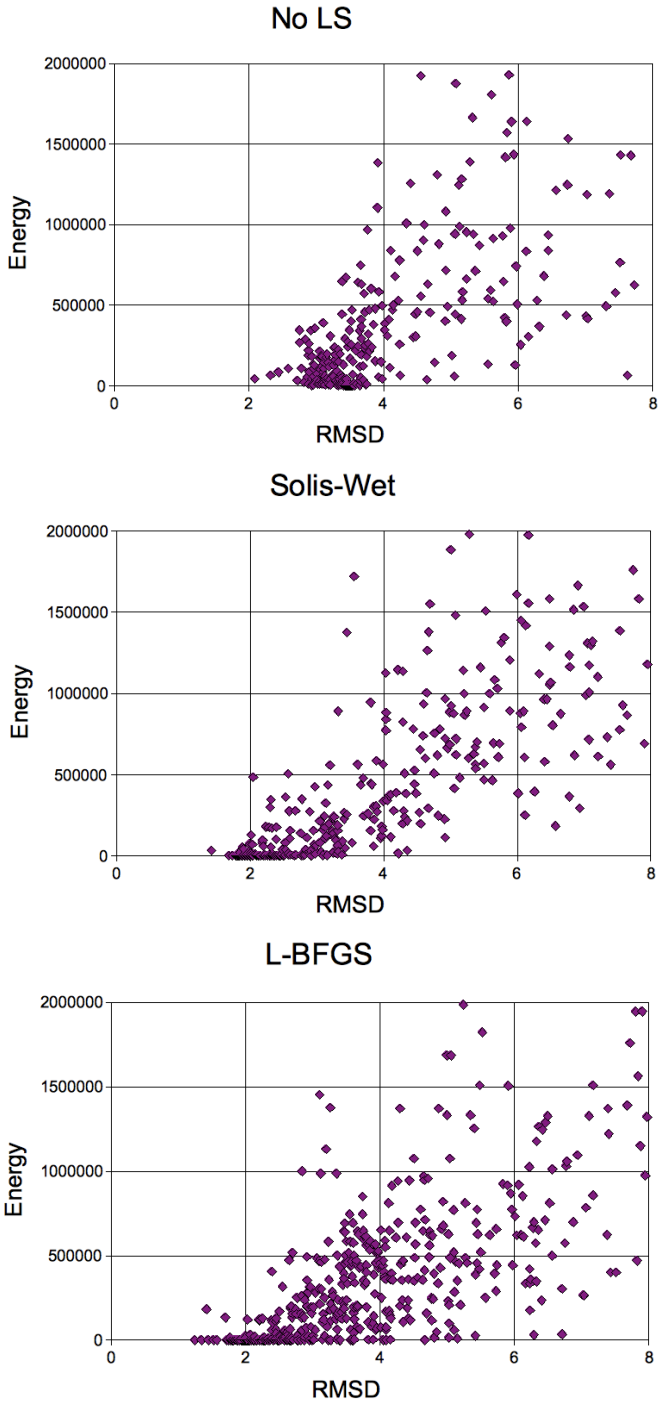


Fig. 1. Scatter plot of an optimization run for the 1adb complex with all variants

Table 4. Full runs for the 1adb and 1hvr instances, sorted by best Energy values

Runs	1adb						1hvr					
	No LS		Solis-Wets		L-BFGS		No LS		Solis-Wets		L-BFGS	
	Energy	RMSD	Energy	RMSD	Energy	RMSD	Energy	RMSD	Energy	RMSD	Energy	RMSD
1	-6.59	1.53	-3.85	1.50	**-18.30**	0.51	-15.79	**0.29**	-13.81	0.65	**-32.43**	0.71
2	-1.51	2.23	-0.28	2.06	-18.25	0.48	-15.58	0.73	-13.70	0.66	-32.42	0.62
3	-1.19	1.81	2.50	1.34	-18.18	0.40	-15.52	0.48	-13.36	0.65	-32.34	0.62
4	0.04	1.47	7.30	1.98	-18.15	**0.30**	-15.30	0.67	-13.26	0.68	-32.31	0.70
5	8.25	2.21	8.67	2.75	-18.08	0.45	-15.22	0.71	-13.16	0.53	-32.30	0.58
6	9.42	2.35	9.44	3.50	-18.27	0.41	-14.42	0.86	-13.11	0.61	-32.27	0.55
7	12.08	2.64	9.45	3.18	-18.15	0.55	-14.17	0.89	-12.96	0.66	-32.26	0.55
8	22.59	1.85	14.13	3.28	-17.92	0.49	-13.96	0.75	-12.92	0.66	-32.26	0.60
9	32.85	3.29	14.25	1.40	-17.84	0.39	-13.88	0.84	-12.81	0.66	-32.25	0.59
10	36.87	2.39	15.43	2.10	-17.79	0.48	-13.75	0.85	-12.80	0.88	-32.25	0.69
11	37.38	2.79	16.30	2.40	-17.72	0.49	-13.26	0.80	-12.59	0.57	-32.20	0.63
12	40.45	1.71	16.45	1.39	-17.58	0.44	-12.98	0.92	-12.41	0.73	-32.19	0.69
13	47.21	1.83	17.55	2.76	-17.43	0.42	-12.90	0.89	-12.36	0.86	-32.08	0.56
14	55.59	2.86	24.19	2.32	-15.97	0.89	-12.73	1.01	-12.04	1.23	-32.03	0.50
15	56.95	4.32	43.29	2.21	-15.58	3.71	-12.72	0.93	-11.58	0.72	-31.97	0.56
16	63.86	1.99	49.75	2.22	-15.55	1.36	-12.51	0.77	-11.48	0.76	-31.35	0.72
17	64.04	2.89	59.49	4.12	-15.32	2.94	-12.51	1.18	-11.09	0.98	-32.34	0.60
18	65.29	2.56	76.65	1.91	-13.89	1.59	-12.37	1.20	-10.85	0.93	-32.21	0.65
19	72.67	3.60	88.04	3.19	-13.86	1.61	-12.26	0.94	-10.66	1.12	-32.20	0.56
20	91.42	4.18	93.40	2.29	-12.76	2.16	-9.31	1.05	-10.10	1.16	-32.15	0.58
21	115.83	2.81	98.69	7.91	-12.37	2.39	-8.28	0.95	-10.07	0.94	-32.13	0.58
22	162.20	2.85	100.14	3.24	-12.05	2.27	-7.90	1.06	-9.82	0.86	-31.28	0.67
23	169.17	2.80	101.90	2.58	-11.92	1.90	-7.05	0.93	-9.70	1.16	-32.30	0.57
24	213.00	2.87	137.47	2.76	-11.03	2.78	-5.89	1.06	-9.62	1.06	-32.23	0.58
25	252.60	3.76	154.18	2.08	-9.16	2.85	-4.50	1.52	-7.24	1.38	-32.17	0.60
26	257.58	2.75	225.49	2.14	-8.02	3.92	-2.32	0.93	-5.76	1.30	-32.06	0.58
27	299.62	2.56	301.42	3.69	-7.65	2.80	1.89	1.49	-3.40	1.07	-32.22	0.58
28	423.42	3.42	314.32	3.21	-6.83	2.97	6.15	1.07	-1.21	1.12	-32.20	0.60
29	2900.60	4.21	721.81	3.84	-2.25	3.52	14.49	1.42	-0.57	1.29	-32.07	0.59
30	4558.50	4.85	3053.11	4.42	-2.13	3.83	24.45	1.55	0.61	1.17	-31.44	0.74
Avg	335.34	2.78	192.36	2.79	**-14.00**	**1.64**	-8.67	0.96	-10.13	0.90	**-32.13**	**0.61**
Std	953.64	**0.87**	559.51	1.26	**4.80**	1.26	9.39	0.28	4.07	0.25	**0.28**	**0.06**

The second important information gathered from this table is the relation between the RMSD value attained with the best energy. The lowest energy found is only good if it corresponds to a low RMSD value. For the best energy values in table 2, we present the corresponding RMSD values in the column Best-En. From this column it is possible to conclude that the distribution of the best values is more balanced between the two algorithms with local search methods. The Solis-Wets method has the best values for four instances and the L-BGFS algorithm for three instances. The remaining best value is for the evolutionary algorithm without local search. The first conclusion we can draw from this is that the L-BFGS method is more capable to optimize in terms of energy minimization while the other methods, although with higher energies, can reach solutions with a more similar structure to the optimal case. Nevertheless, the function being optimized is in terms of energy and the L-BFGS method shows that it is capable to reach more efficiently the desired area to search.

This is shown by the scatter plots in figure 1 and the full optimization runs presented in table 4. The plots display for the 1adb complex (without major differences, the same pattern is observed in the remaining instances) the position

of the discovered solutions the relation between energy and RMSD for an optimization run. The closer to the origin of the axis the better. Common patterns can be detected on all the plots: there is a clear approximation flow to the origin and there is a concentration between RMSD values of 2 and 4 with an energy level below 500000. The main differences between the plots are: 1) the degree of the concentration is higher for the L-BFGS method; 2) the majority of the initial solutions for the L-BFGS method have a RMSD below 8, thus they are presented in the plot; 3) although the Solis-Wet algorithm shows a small focus of solutions in the RMSD level below 2, this effect is stronger with the L-BFGS method. The main information from the these plots is that an optimization run with the L-BFGS method drives more solutions with lower energy and RMSD values closer to the desired point.

A perusal of table 4 helps to consolidate this view. The table presents the data from two complexes, 1adb and 1hvr, which show the best energy and corresponding RMSD for the 30 runs, for the three tested algorithms. The runs are sorted by the energy values. Important differences between the evolutionary algorithms are found. First, the L-BFGS method is able to consistently reach good energy values. This is very clear when compared to any of the other two methods, especially for the later runs. The L-BFGS method still provides good energy values while the Solis-Wets method and no local search display unacceptable energy values. In addition, the same kind of behavior is displayed when observing the RMSD columns. Although less strong in comparison to energy, the L-BFGS method is able to provide lower RMSD values for a longer number of runs. In the case of the 1hvr complex, the L-BFGS algorithm presents values always below 1.0 while for the remaining algorithms this is not true. Furthermore, for the 1adb complex, L-BFGS is the only method capable of attaining RMSD values below 1.0 and for almost half the runs (14 out of 30). Similar patterns are also observed in the remaining complexes optimization runs.

4 Conclusions

We investigated the use of local search methods when applied to the molecular docking problem. Previous research on this topic indicated that the most widely used local search method for docking could be inefficient and, with carefully selected genetic operators, an evolutionary algorithm is sufficient [3]. Our investigation partially supports this view. Although the Solis-Wets algorithm might not be the most suitable local search component in an evolutionary algorithm, other local search methods could prove to be efficient. In this paper, we proposed a standard generational evolutionary algorithm hybridized with the L-BFGS method, a powerful quasi-Newton conjugate gradient method. Results show that an evolutionary algorithm with this method as local search is superior to previous approaches [6,3]. The optimization results are considerably better in terms of energy and RMSD values. The differences between this algorithm and the other tested approaches are statistically significant.

References

1. Neumaier, A.: Molecular modeling of proteins and mathematical prediction of protein structure. SIAM Review 39, 407–460 (1997)
2. Thomsen, R.: Protein-ligand docking with evolutionary algorithms. In: Fogel, G.B., Corne, D.W., Pan, Y. (eds.) Computational Intelligence in Bioinformatics, pp. 169–195. Wiley-IEEE Press (2008)
3. Thomsen, R.: Flexible ligand docking using evolutionary algorithms: investigating the effects of variation operators and local search hybrids. Biosystems 72, 57–73 (2003)
4. Tavares, J., Tantar, A.A., Melab, N., Talbi, E.G.: The influence of mutation on protein-ligand docking optimization: a locality analysis. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199. Springer, Heidelberg (2008)
5. Tavares, J., Melab, N., Talbi, E.G.: An empirical study on the influence of genetic operators for molecular docking optimization. Technical Report RR-6660, INRIA Lille - Nord Europe research Centre (2008)
6. Morris, G.M., Goodsell, D.S., Halliday, R.S., Huey, R., Hart, W.E., Belew, R.K., Olson, A.J.: Automated docking using a lamarckian genetic algorithm and and empirical binding free energy function. Journal of Computational Chemistry 19, 1639–1662 (1998)
7. Dixon, J.S.: Flexible docking of ligands to receptor sites using genetic algorithms. In: Proc. of the 9th European Symposium on Structure-Activity Relationships, Leiden, The Netherlands, pp. 412–413. ESCOM Science Publishers (1993)
8. Moitessier, N., Englebienne, P., Lee, D., Lawandi, J., Corbeil, C.: Towards the development of universal, fast and highly accurate docking/scoring methods: a long way to go. British Journal of Pharmacology 153, 1–20 (2007)
9. Liu, D.C., Nocedal, J.: On the limited memory method for large scale optimization. Mathematical Programming B, 503–528 (1989)
10. Tavares, J., Tantar, A.A., Melab, N., Talbi, E.G.: The impact of local search on protein-ligand docking optimization. In: Proceedings of the 8th International Conference on Hybrid Intelligent Systems (2008)
11. Nelder, J.A., Mead, R.: A simplex method for function minimization. Computer Journal 7, 308–313 (1965)

A Comparison of Genetic Algorithms and Particle Swarm Optimization for Parameter Estimation in Stochastic Biochemical Systems

D. Besozzi[1], P. Cazzaniga[2], G. Mauri[2], D. Pescini[2], and L. Vanneschi[2]

[1] Università di Milano. Dipartimento di Informatica e Comunicazione
Via Comelico 39, 20135 Milano, Italy
besozzi@dico.unimi.it
[2] Università di Milano-Bicocca. Dipartimento di Informatica, Sistemistica e Comunicazione
Viale Sarca 336, 20126 Milano, Italy
cazzaniga/mauri/pescini/vanneschi@disco.unimib.it

Abstract. The modelling of biochemical systems requires the knowledge of several quantitative parameters (e.g. reaction rates) which are often hard to measure in laboratory experiments. Furthermore, when the system involves small numbers of molecules, the modelling approach should also take into account the effects of randomness on the system dynamics. In this paper, we tackle the problem of estimating the unknown parameters of stochastic biochemical systems by means of two optimization heuristics, genetic algorithms and particle swarm optimization. Their performances are tested and compared on two basic kinetics schemes: the Michaelis-Menten equation and the Brussellator. The experimental results suggest that particle swarm optimization is a suitable method for this problem. The set of parameters estimated by particle swarm optimization allows us to reliably reconstruct the dynamics of the Michaelis-Menten system and of the Brussellator in the oscillating regime.

1 Introduction

The modelling of biological systems requires the knowledge of many numerical factors, like molecular species concentrations and reaction rates, which represent an indispensable quantitative information to perform computational investigations of the system behavior. Unfortunately, the experimental values of these factors are often not available or inaccurate, since carrying out their measurements *in vivo* can be tangling or even impossible [16]. In a few cases, the values of some parameters of a given system can be estimated either from *in vitro* experiments (by fitting the dynamics derived through equations based on mass-action law against the concentration time series that result from these measurements), or by assuming some analogies with similar processes or organisms for which more experimental data are available. In general, the lack and the inaccuracy of these information bring about the challenging problem of assigning the correct values to all parameters in order to reproduce the expected dynamics in the best possible way. Optimization methods can be used to tackle the calibration problem of *parameter estimation* by minimizing a cost function (e.g. a distance measure)

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 116–127, 2009.

which quantitatively defines how good is the system behavior using the predicted values, with respect to the experimental dynamics. Several global optimization techniques (see [16,14] and references therein) have already been adopted for parameter estimation of biochemical and biological systems. A peculiarity of our approach, with respect to other methods previously investigated for this problem, is that it embeds the (forward) problem of performing *stochastic* simulations of a system dynamics, into the (inverse) problem of estimating its unknown parameters. This leads to the development of a method that exploits the outcome of stochastic simulation algorithms – usually used to generate the temporal dynamics of the system's chemicals – to effectively define the fitness function used by the optimization techniques here investigated. In particular, we compare the performances of genetic algorithms (GAs) and particle swarm optimization (PSO) for the parameter estimation of two simple biochemical schemes which are well representative of the dynamics of many biological systems: a basic catalytic kinetics (the Michaelis-Menten system), a sustained oscillating behavior and a dumped oscillating behavior (both based on the Belousov-Zhabotinskii reaction).

The motivation for considering stochastic processes comes from several experimental investigations, which have evidenced the role of "biological noise" in living cells that can drive cellular processes, involving small populations of reactant species, into stochastic behaviors [7,2]. Works of this type show the inadequacy of the traditional deterministic modelling to describe phenomena such as signalling pathways, especially when gene expression is involved, therefore supporting the need for stochastic modelling approaches. Systems of biochemical reactions have been traditionally analyzed by means of deterministic methods, corresponding to sets of coupled ordinary differential equations (ODEs), where one equation is drawn for each molecular species involved in the system, specifying all the transformations this species undergoes as either a reagent or a product (this leads to the so called *reaction-rate equation*). On the other hand, a different investigation of biochemical systems can be carried out by means of computational simulations, as molecular dynamics methods, where one follows the discrete time evolution of the system, and the quantities of the species are given by the copy numbers of their molecules. One of the pioneering work behind the latter approach is the stochastic simulation algorithm (SSA), proposed by Gillespie in [8]. In the present work we consider the *tau leaping* stochastic simulation algorithm [4], which is an approximate and faster version of SSA. In both SSA and tau leaping, the probabilistic choice of the tossed reactions and of their application time depends on both the type of the reactions and the current amounts of reagents inside the volume (i.e. the state of the system). Furthermore, they share the characteristic that repeated (independent) executions of the algorithm will produce different temporal dynamics, even starting from the same initial configuration, thus reflecting the inherent noise (this is in contrast to ODEs, which always reproduce exactly the same dynamics starting from the same initial conditions). In practice, in order to define the fitness function, which is one of the main problems in applying heuristics like GAs or PSO to this kind of problems, we evaluate the individuals by considering, for each molecular species in the system, the similarity/dissimilarity between two dynamics: the target (experimentally known) dynamics on one side, and the estimated dynamics on the other side. The first dynamics

is here reconstructed by solving the reaction-rate equation of the biochemical system, while the latter is generated by running (some executions of) the tau leaping algorithm.

The paper is structured as follows. In Section 2 we provide some basic notions of biochemical reactions, and describe the two simple systems (Michaelis-Menten reaction and the Brussellator) that we will consider for parameter estimation. In Section 3 we present the versions of GAs and PSO that we have used and, in particular, we give a suitable definition of fitness function, taking into account the stochastic nature of the estimated dynamics. The experimental results are presented in Section 4, while further applications of these methods and some open problems are discussed in Section 5.

2 Systems of Biochemical Reactions

Any generic (bio)chemical reaction can be written in the form $r : \sum_{i=1}^{N_1} \alpha_i R_i \rightarrow \sum_{j=1}^{N_2} \beta_j P_j$, where $R_1, \ldots, R_{N_1}$ are distinct reactant molecular species and $P_1, \ldots, P_{N_2}$ are distinct products, for some $N_1, N_2 \geq 0$. The non-negative integers α_i, β_j are the *stoichiometric coefficients* of the reaction r; they specify how many molecules of each reagent species are necessary to trigger the reaction, and how many product molecules are formed after the reaction has occurred. Each reaction is also characterized by a numerical factor, called the *rate constant*, which determines – together with the amount of reagents – the rate, or velocity, of the reaction itself. When a system of reactions is analyzed by means of a stochastic algorithm, the rate of each reaction is determined by the so called stochastic constant (usually denoted by c, and having unit of a reciprocal time), a value that encompasses the physical and chemical properties of the reaction. The *kinetics* of a reaction is influenced by many factors, such as temperature, pressure, amounts of reactants, molecular crowding, etc. As a consequence, the experimental determination of rate constants for a given biochemical system is not a trivial task, and the situation gets even harder when considering more complex biological systems, like metabolic pathways or cellular processes in general. For further notions about biochemical reactions and chemical kinetics we refer to [5,15].

Michaelis-Menten system. The first chemical system we consider, the *Michaelis-Menten* (MM) kinetics, describes the catalytic transformation of one-reacting substrate (denoted by S) into a final product (P) mediated by an enzyme (E), passing through the (relatively fast) reversible formation of the enzyme-substrate intermediate complex (ES). The role of E is to lower the energy required by S for its interconversion to P; the enzymatic kinetics assumes that S and E are in a fast equilibrium with the complex they form, which then dissociates to yield the product while releasing the enzyme free. The set of chemical reactions corresponding to MM is: $E + S \xrightarrow{c_1} ES, ES \xrightarrow{c_2} E+S, ES \xrightarrow{c_3} E+P$. The initial amounts of the substrate and the enzyme used to generate the target dynamics, shown (with solid line) in Fig. 5(a), are $S = 1000$, $E = 750$ molecules, while the stochastic constant values (that we want to estimate with GAs and PSO) are $c_1 = 2.5 \cdot 10^{-3}, c_2 = 0.1, c_3 = 5$.

Brussellator system. The second chemical system we consider, called *Brussellator* [17], is a simplified scheme for the Belousov-Zhabotinskii reaction, a family of inorganic redox reaction systems that exhibit macroscopic temporal oscillations and spatial

patterns formation. This theoretical scheme is recognized as the prototype of nonlinear oscillating (open and well-stirred) systems, proving the significance and variety of both spatial organizations and complex rhythms occurring in many biological systems. Here, we give a description of the Brussellator that slightly differs from the original one: with respect to the formulation given in [17], we leave out the presence of two products (since they are not directly involved in the formation of the oscillating limit cycle), and we consider the following set of reactions: $A \xrightarrow{c_1} X, B + X \xrightarrow{c_2} Y, 2X + Y \xrightarrow{c_3} 3X, X \xrightarrow{c_4} \lambda$, where A, B are two chemicals that are given as input and always kept at a constant amount, X, Y are the intermediate product chemicals that exhibit oscillations, and λ represents the degradation of species X. The initial amounts of molecules used for generating the dynamics are $A = X = 200$, $B = 600$ and $Y = 300$. We consider two distinct sets of values for the reaction constants $\{c_1, c_2, c_3, c_4\}$ to be estimated using GAs and PSO. The set $\{1, 5 \cdot 10^{-3}, 2.5 \cdot 10^{-5}, 1.5\}$ gives rise to sustained and periodic oscillations in the species X, Y, while the set $\{1, 5 \cdot 10^{-3}, 2.5 \cdot 10^{-4}, 1.5\}$ gives rise to dumped oscillations and quickly drives the system dynamics to a steady-state. The corresponding dynamics are shown (with solid and dashed lines) in Fig(s). 5(b) and 5(c), respectively.

3 Genetic Algorithms and Particle Swarm Optimization

In this paper, we compare the performances of two different optimization algorithms: GAs [9] and PSO [11]. This choice is motivated by the fact that the presented applications are an example of dynamic optimization problems, meaning that the fitness function may change (more precisely, each individual can have slightly different fitness values each time it is evaluated) and a number of contributions exist about the use of GAs and PSO for this kind of problems (see, e.g., [12,10]).

Both GAs and PSO formulations used in this work evolve individuals of the same shape: n-length vectors of floating point numbers, where n is the number of rate constants to optimize. The experimental settings and the related choices that we have done for GAs and PSO are described below.

Genetic Algorithms. The most commonly used GAs formulation evolves fixed length strings over a finite alphabet, while in our case each allele can contain any floating point value from a limited range. This GAs version is often called *real-valued* or *real-coded* GAs (see, e.g., [19]). Even though we are aware that many sophisticated genetic operators for real-coded GAs have recently been defined (like, for instance, Laplace Crossover, Makinen, Periaux and Toivanen Mutation, Non-Uniform Mutation [6]), in this study, that represents a first step in our investigation, we prefer to use simpler operators, like the ones originally defined in [19], more precisely: *one point* and *average crossover*, *gaussian mutation*, *range mutation* and *reinitialization*.

One point crossover is similar to the standard GAs crossover defined by Holland in [9]: parents are aligned, one crossover point is selected and substrings are inverted to generate offspring. Average crossover returns one offspring that contains at each position the average values of the parents chromosomes. In this work, crossover is executed with a 0.95 rate. In case crossover is not executed, parents are copied in the next

population with no modification (reproduction). Otherwise, one operator among one point crossover and average crossover is chosen with uniform probability distribution.

Gaussian mutation perturbs an allele with a number drawn from a normal distribution with mean μ equal to the current value of that allele and $\sigma = 0.05 \cdot \mu$. Range mutation increments or decrements an allele of a prefixed quantity (equal to 5% of its admissible range size in this work). Reinitialization changes the value currently contained in an allele with an uniformly distributed random number in the admissible range. For each individual in the spring, mutation is applied to each allele with probability $1/n$, where n is its length. If an allele has to be mutated, one operator among gaussian mutation, range mutation or reinitialization is chosen with uniform probability distribution.

The other parameters that we have used are: population size of 100 individuals; maximum number of generations equal to 100; tournament selection of size 5; elitism (i.e. copy of the best individual unchanged in the next population at each generation).

Particle Swarm Optimization. PSO [11] is a population based optimization heuristic, inspired by social behavior of bird flocking or fish schooling. In PSO the potential solutions, called particles, are identified by their coordinates in the problem space and are also characterized by a velocity that allows them to update their current positions. The PSO concept consists in changing, at each iteration, the velocity of each particle towards some attractors, typically the global and local best positions that have been found so far. The swarm behavior is influenced by two parameters, C_1 and C_2, that control global exploration and local exploitation, and try to prevent the particles from prematurely converging to local minima.

Starting from the work presented in [18], several papers have been recently published which aim at improving the performances of the PSO with different settings, by focusing on the optimization of parameters, such as constants C_1, C_2 (see, e.g. [3,1,18]). Among these contributions, we are particularly interested in [3] where the co-evolutionary PSO algorithm, aimed at optimizing the values of constants C_1 and C_2 first proposed by Miranda and Fonseca in [13], is investigated. Authors of [3] study this co-evolutionary PSO version for a large set of well-known theoretically hand-tailored problems and hint that it may outperform standard PSO for complex problems. For this reason, in this paper we study two versions of the PSO: the canonical PSO formulation (indicated with PSO1 from now on) and the co-evolutionary Miranda and Fonseca model (PSO2 from now on).

The other experimental settings that we have used for both these PSO versions are: swarm size of 20 particles; maximum number of iterations equal to 500 (so doing, GAs and PSO will have executed the same number of fitness evaluations at the end of a run); the inertia weight w has been linearly decremented from 0.9 to 0.2 with gaussian noise with average equal to the current w value and $\sigma = 0.05$ (as reported in [18]). In addition, in PSO2 we add a gaussian noise to C_1 and C_2 with $\sigma = 0.1$. For both PSO versions, we have considered velocity ranges of half the size of the rate constants ranges and when a particle reaches a bound of the admissible interval, its velocity is halved and its direction inverted.

Fitness Function. We give here a novel definition of fitness based on the idea that, in order to optimize the rate constants of a stochastic biochemical system, we have

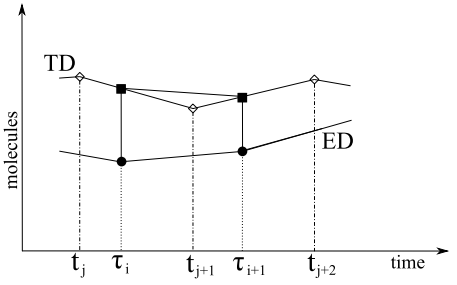


Fig. 1. Area evaluation between TD and ED series for the fitness function

to compare the given experimental outcome (the target dynamics) with the estimated dynamics generated by the tau leaping algorithm, which will run using the parameter values codified in the individuals of GAs or PSO. To this aim, we have to manage some troublesome properties, hereby discussed, that are inherent to stochastic simulations. The fitness of each individual will be evaluated by calculating the *area* between the target dynamics (TD) and the estimated dynamics (ED) of each molecular species with known behavior. In this work, pseudo-experimental TD are generated by means of an ODEs solver, according to the values assumed for the rate constants and reported in Section 2. The conversion between the deterministic and the stochastic parameters has been done according to the relations given in [8], and assuming a reaction volume equal to $1.6 \cdot 10^{-17} L$ for MM and to $3.2 \cdot 10^{-22} L$ for the Brussellator.

To be more precise, the fitness function is defined as follows. Let $L \subseteq \{1, \ldots, M\}$ be the set of molecular species whose dynamics is assumed to be experimentally known, t_0 and t_N the initial and final time instants of the given TD. We denote by $x'_l(t_0), \ldots, x'_l(t_N)$ the TD time series of species l, $l \in L$, and by $x_l(\tau_0), \ldots, x_l(\tau_{\widetilde{N}})$ its ED time series, where $\tau_0 = t_0$ and $\tau_{\widetilde{N}} \cong t_N$. Note that, in general, all the time instants (except the initial one) of the ED series will be distinct from those sampled in the TD series, since the two methods use different time samplings and, above all, each simulation by means of tau leaping generates a different time series (hence, it does not correspond to the constant-step temporal sampling of TD). In addition, the time interval between any couple of consecutive time instants of the ED series will be generally distinct from every other time interval in the same series. As the evaluation of the fitness function requires to determine the difference between the TD and the ED, we need to pick up couples of values, one in the TD series and the other one in the ED series, that correspond to an identical time instant. Therefore, for each simulated time instant τ_i, $i = 0, \ldots, \widetilde{N}$ (filled circles in Fig. 1), we consider the two consecutive time instants in the TD series t_j, t_{j+1}, $j = 0, \ldots, N-1$ (empty diamonds in Fig. 1), which satisfy the following conditions: (1) $t_j \leq \tau_i \leq t_{j+1}$; (2) there exist no other time instants t'_j, t'_{j+1}, $j = 0, \ldots, N-1$, such that $t_j \leq t'_j \leq \tau_i \leq t'_{j+1} \leq t_{j+1}$. Then, by performing a linear interpolation between the given TD series values $x'_l(t_j)$ and $x'_l(t_{j+1})$, we derive the element $x'_l(\tau_i)$ (filled squares in Fig. 1) in the TD series corresponding to the element $x_l(\tau_i)$ in the ED series, and we evaluate their distance, $|x'_l(\tau_i) - x_l(\tau_i)|$. We then compute the fitness f_z (for each independent execution z of the tau leaping algorithm) by summing up,

for each simulated time instant τ_i and for each molecular species $l \in L$, the areas of the trapezoids (thick lines in Fig. 1) having as basis the distances $|x'_l(\tau_i) - x_l(\tau_i)|$ and $|x'_l(\tau_{i+1}) - x_l(\tau_{i+1})|$, and as height the length of the time interval $[\tau_i, \tau_{i+1}]$, that is: $f_z = \sum_{i=1}^{\tilde{N}-1} \sum_{l \in L} \frac{1}{2}(|x'_l(\tau_i) - x_l(\tau_i)| + |x'_l(\tau_{i+1}) - x_l(\tau_{i+1})|)(\tau_{i+1} - \tau_i)$. Finally, the fitness function is defined as $f = \frac{1}{Z} \sum_{z=1}^{Z} f_z$, where Z is the total number of independent runs of the tau leaping algorithm executed for each individual of GAs and PSO.

4 Experimental Results

In this section we discuss the application of our proposed GAs and PSO versions, presented in Section 3, for the estimation of rate constants of the systems described in Section 2, for which we have used the following admissible ranges: $c_1 \in [2.5 \cdot 10^{-5}, 2.5 \cdot 10^{-1}], c_2 \in [1 \cdot 10^{-3}, 10], c_3 \in [5 \cdot 10^{-2}, 5 \cdot 10^{2}]$ for the MM system (three further ranges have been tested for this system, obtaining qualitatively similar results to the ones presented here); $c_1 \in [0.1, 10], c_2 \in [5 \cdot 10^{-4}, 5 \cdot 10^{-2}], c_3 \in [2.5 \cdot 10^{-6}, 2.5 \cdot 10^{-4}], c_4 \in [0.15, 15]$ for the Brussellator system with oscillating regime and $c_1 \in [0.1, 10], c_2 \in [5 \cdot 10^{-4}, 5 \cdot 10^{-2}], c_3 \in [2.5 \cdot 10^{-5}, 2.5 \cdot 10^{-3}], c_4 \in [0.15, 15]$ for the Brussellator system with dumped oscillations. Each range has been chosen such that the lower and upper bounds are two (resp. one) orders of magnitude below and above the target value for MM (resp. Brussellator).

For both GAs and PSO, and for each individual in the population, fitness is evaluated by performing a fixed number Z of independent executions of the tau leaping stochastic algorithm, chosen according to the system dynamics. For each system we report the Average Best Fitness (ABF) against fitness evaluations, and the accumulated number of successful runs at each considered value of the fitness evaluations. Given that it is unlike to obtain a fitness equal to zero (see discussion in Section 5), a run has been considered successful if at least one individual (that we improperly call an optimal solution) has been found with a fitness value smaller than $1.1 \cdot \overline{f}$, where $\overline{f}$ is the best fitness value obtained among any one of the three algorithms and any one of the executed runs (50 in the following examples).

Michaelis-Menten. Figure 2 presents the experimental results returned by GAs, PSO1 and PSO2 for the MM system. These results have been obtained with 50 independent runs of GAs, PSO1, PSO2, and $Z = 10$ tau leaping simulations. In Fig. 2(a) we report the ABF and in Fig. 2(b) we report the number of successful runs. Figure 2(a) shows that GAs have returned a better ABF than the two PSO variants at each value of the fitness evaluations that we have studied, while Fig. 2(b) shows that PSO1 has obtained the largest number of successful runs, in particular after 6,000 fitness evaluations. This different behavior between GAs and PSO1 hints that PSO1 has found optimal solutions more frequently than GAs, but when an optimal solution has not been found, GAs have returned individuals of better quality. We also point out that the ABFs of PSO1 and PSO2 are similar to each other (in particular after 9,000 fitness evaluations), while PSO1 has been able to find optimal solutions more often than PSO2.

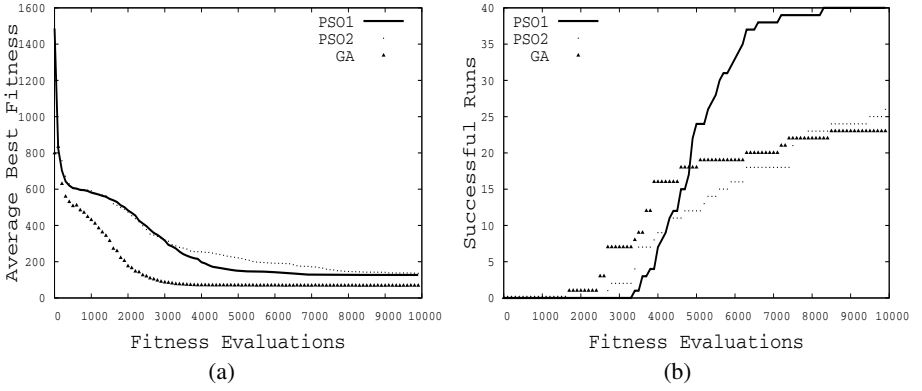


Fig. 2. Experimental results returned by GAs, PSO1 and PSO2 for the Michaelis-Menten system, obtained by executing 50 independent runs of each algorithm. (a): Average Best Fitness against fitness evaluations. (b): Accumulated number of successful runs against fitness evaluations.

In Fig. 5(a) we report the dynamics of the MM system generated by tau leaping using the constants found by the best solution generated by PSO1 over the considered 50 runs, which are $c_1 = 0.00245, c_2 = 0.02691, c_3 = 5.03552$. We compare this dynamics (dots) with the target curves (solid lines), pointing out that it very well approximates the targets for each one of the four chemicals involved in the reaction (E, S, ES, P).

Brussellator with oscillating regime. Figure 3 reports the experimental results returned by GAs, PSO1 and PSO2 for the Brussellator system with oscillating regime. In particular, Fig. 3(a) reports the ABF and Fig. 3(b) reports the number of successful runs. These results have been obtained with 50 independent runs of GAs, PSO1, PSO2; in this case, only one tau leaping execution ($Z = 1$) has been performed because, if we

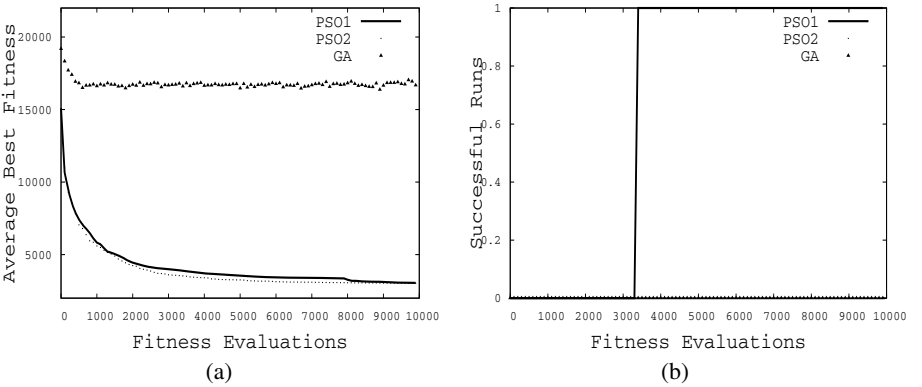


Fig. 3. Experimental results returned by GAs, PSO1 and PSO2 for the Brussellator system with oscillating regime, obtained by executing 50 independent runs of each algorithm. (a): Average Best Fitness against fitness evaluations. (b): Accumulated number of successful runs against fitness evaluations.

execute more than one run and we calculate their average behaviors, we would risk to flatten the oscillations that characterize this dynamics.

In this case, it is clear that PSO outperforms GAs both from the ABF and success rate viewpoints. In particular, we point out that neither GAs nor PSO2 have been able to find an optimal solution in none of the 50 independent executions that we have performed, while PSO1 has found an optimal solution in only a single run after a number of fitness evaluations approximately equal to 3,500. We conclude that PSO1 can be considered the most suitable algorithm, among the ones that we have studied, for this particular Brussellator dynamics.

Figure 5(b) reports the dynamics (dots) over one period of the Brussellator system with oscillating regime that we have obtained using the constants found by the best solution generated by PSO1 over the 50 runs that we have executed. We also observe that the target behavior (lines) has been reliably approximated with the estimated constants, that are $c_1 = 1.23966, c_2 = 0.00634, c_3 = 4.37171 \cdot 10^{-5}, c_4 = 2.71063$.

Brussellator with dumped oscillations. The experimental results returned by GAs, PSO1 and PSO2 for the Brussellator system with steady-state attractor are reported in Fig. 4. As for the other cases, in Fig. 4(a) we report the ABF and in Fig. 4(b) the accumulated number of successful runs, obtained with 50 independent runs of GAs, PSO1, PSO2, and $Z = 5$ tau leaping simulations.

Also in this case, as for the Brussellator system with oscillating regime, PSO clearly outperforms GAs both for ABF and success rate. In this case, PSO1 and PSO2 show a similar behavior for both these statistics. GAs have not been able to find an optimal solution in none of the 50 independent executions that we have performed, while both PSO variants have found an optimal solution in all the 50 runs after a number of evaluations approximately equal to 5,500.

Given that PSO1 has been able to obtain a success rate equal to 1 with a slightly lower number of fitness evaluations than PSO2, also in this case we report the

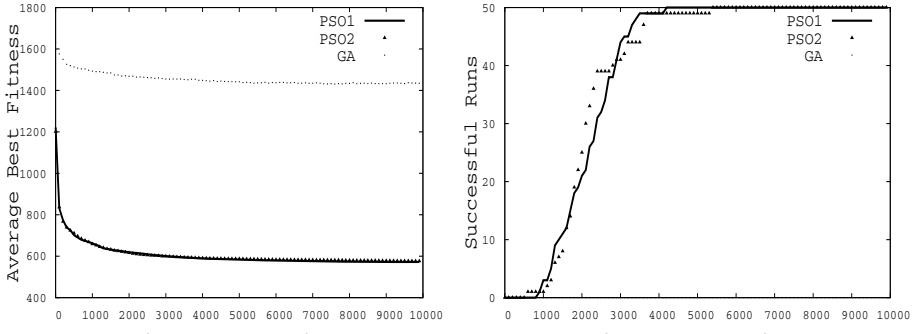


Fig. 4. Experimental results returned by GAs, PSO1 and PSO2 for the Brussellator system with dumped oscillations, obtained by executing 50 independent runs of each algorithm. (a): Average Best Fitness against fitness evaluations. (b): Accumulated number of successful runs against fitness evaluations.

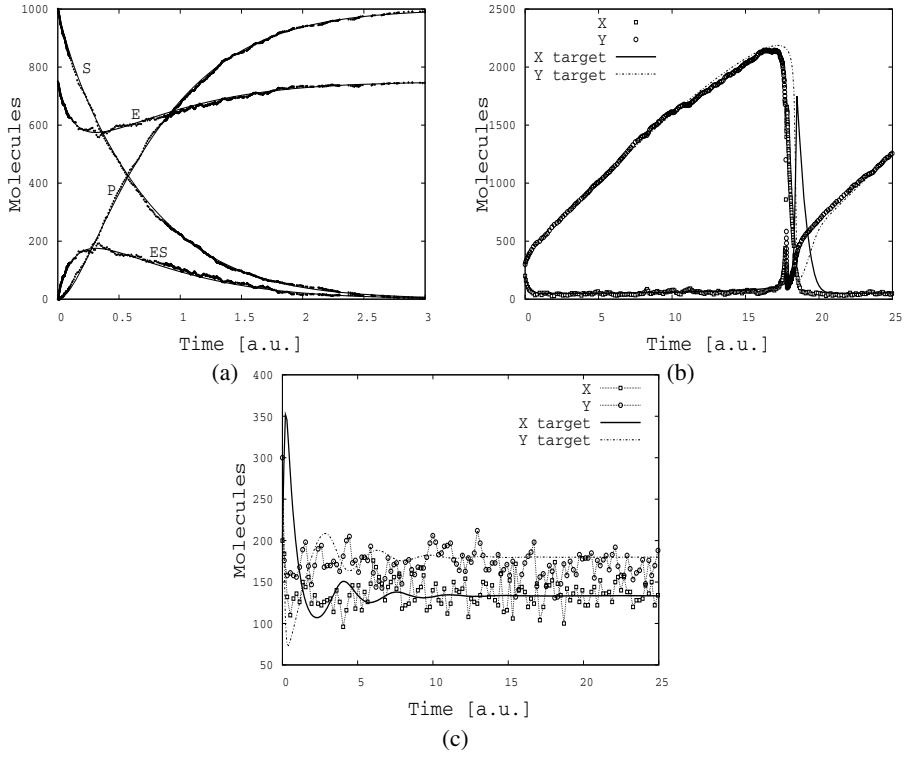


Fig. 5. Dynamics of the studied systems using the constants found by the best solution generated by PSO1. (a): Michaelis-Menten system; (b): Brussellator with oscillating regime; (c): Brussellator with dumped oscillations.

dynamics obtained using the constants contained in the best solution found by PSO1 ($c_1 = 9.9571, c_2 = 0.00616, c_3 = 0.00033, c_4 = 15$). These dynamics are shown in Fig. 5(c). This time the estimated dynamics (dots) of the two chemicals do not approximate targets (lines) in a satisfactory way. In particular, we evidence that the estimated dynamics approximates the target at the steady-state, while the first part of the target dynamics, with dumped oscillations, is not reliably reconstructed. We hypothesize that this problem could be solved by assigning to the second (non-oscillatory) part of the dynamics a lower weight than to the first (oscillatory) one in the fitness calculation.

5 Discussion

The problem of estimating the rate constants of two well known biochemical systems has been tackled in this paper, using GAs and two different versions of PSO. The experimental results that we have obtained hint that canonical PSO is a suitable optimization method for this problem, since it outperforms both GAs and the PSO version that co-evolves parameters C_1 and C_2. The values of the constants returned by PSO

have allowed us to faithfully reconstruct the behavior of the MM system and of the oscillating Brussellator. Nevertheless, for the dumped Brussellator even the best set of constants found by PSO has not allowed us to reconstruct the target dynamics in a suitable way. Thus, even though interesting, the presented results deserve further investigations.

Some possible improvements to the proposed optimization methods include the use of further GAs genetic operators, for instance, Laplace Crossover, Makinen, Periaux and Toivanen Mutation, Non-Uniform Mutation [6], and other PSO models, like PSO with structured populations, with various different neighborhood structures or with more than two basins of attraction (in addition to the local and global best positions).

Another possible improvement concerns the fitness function. The definition of a suitable fitness function is an intrinsically difficult task for parameter estimation of biochemical systems, for several reasons. First of all, if we exploit stochastic simulation algorithms for the fitness calculation, then it is very unlike to reach the (ideal) value of zero because of the stochastic fluctuations of the estimated time series. Moreover, the same individual will generally have different fitness values each time it is evaluated. These drawbacks could be mitigated by, for instance, executing many times the tau leaping algorithm and calculating averages, but this might be very time consuming in some cases and it could raise serious problems when the target dynamics shows an oscillating behavior (averages may flatten oscillations).

In addition, regarding the fitness function, our choice of using tau leaping to generate the dynamics corresponding to a given set of constants (and thus to evaluate the fitness of an individual) has suggested us to consider the area between the target and the estimated curves, instead of calculating their point-to-point mutual distance. In fact, since the tau leaping algorithm samples points at arbitrary time instants, we have to deal with irregular timing patterns; furthermore, some portions of the dynamics might be more frequently sampled than others. If we use the point-to-point distance to calculate fitness, all sampled points would receive the same importance, thus overrating the dynamics portions that have been more frequently sampled. On the other hand, the area calculation allows us to better evaluate the dynamics also in regions that have not been sampled by tau leaping, and thus approximate more faithfully the (ideal) situation in which we have information for each possible time instant.

To try and solve these problems, this fitness definition could be further improved, for instance, in two ways. The first consists in partitioning the time axis into subintervals, and performing the optimization by incrementally extending the time interval considered for the fitness calculation. The second consists in giving different weights to regions of the dynamics that show distinct behaviors, according to some previous knowledge of the system. Finally, a different weight could be assigned to the various species involved in the chemical reactions under consideration according, for instance, to the relevance they hold within the system.

References

1. Arumugam, M.S., Rao, M.V.C.: On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (rms) variants for computing optimal control of a class of hybrid systems. Journal of Applied Soft Computing 8, 324–336 (2008)

2. Blake, W.J., Kaern, M., Cantor, C.R., Collins, J.J.: Noise in eukaryotic gene expression. Nature 422(6932), 633–637 (2003)
3. Cagnoni, S., Vanneschi, L., Azzini, A., Tettamanzi, A.G.B.: A critical assessment of some variants of particle swarm optimization. In: Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G.A., Drechsler, R., Ekárt, A., Esparcia-Alcázar, A.I., Farooq, M., Fink, A., McCormack, J., O'Neill, M., Romero, J., Rothlauf, F., Squillero, G., Uyar, A.Ş., Yang, S. (eds.) EvoWorkshops 2008. LNCS, vol. 4974, pp. 565–574. Springer, Heidelberg (2008)
4. Cao, Y., Gillespie, D.T., Petzold, L.R.: Efficient step size selection for the tau-leaping simulation method. The Journal of Chemical Physics 124(4), 044109 (2006)
5. Connors, K.A.: Chemical Kinetics. The Study of Reaction Rates in Solution. VCH Publishers, Inc., New York (1990)
6. Deep, K., Thakur, M.: A new crossover operator for real coded genetic algorithms. Applied Mathematics and Computation 188(1), 895–911 (2007)
7. Fedoroff, N., Fontana, W.: Small numbers of big molecules. Science 297(5584), 1129–1131 (2002)
8. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry 81(25), 2340–2361 (1977)
9. Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)
10. Janson, S., Middendorf, M.: A hierarchical particle swarm optimizer for dynamic optimization problems. In: Raidl, G.R., Cagnoni, S., Branke, J., Corne, D.W., Drechsler, R., Jin, Y., Johnson, C.G., Machado, P., Marchiori, E., Rothlauf, F., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2004. LNCS, vol. 3005, pp. 513–524. Springer, Heidelberg (2004)
11. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948. IEEE Computer Society, Los Alamitos (1995)
12. Maravall, D., de Lope, J.: Multi-objective dynamic optimization with genetic algorithms for automatic parking. Soft. Computing 11(3), 249–257 (2006)
13. Miranda, V., Fonseca, N.: Epso best-of-two-worlds meta-heuristic applied to power system problems. In: Proceedings IEEE Congress on Evolutionary Computation 2002, pp. 1080–1085. IEEE Computer Society Press, Los Alamitos (2002)
14. Moles, C.G., Mendes, P., Banga, J.R.: Parameter estimation in biochemical pathways: A comparison of global optimization methods. Genome Research 13(11), 2467–2474 (2003)
15. Nelson, D.L., Cox, M.M.: Lehninger Principles of Biochemistry, 4th edn. W. H. Freeman, New York (2004)
16. Reinker, S., Altman, R.M., Timmer, J.: Parameter estimation in stochastic biochemical reactions. Systems Biology, IEEE Proceedings 153, 168–178 (2006)
17. Tyson, J.J.: Some further studies of nonlinear oscillations in chemical systems. The Journal of Chemical Physics 58, 3919–3930 (1973)
18. Del Valle, Y., Venayagamoorthy, G.K., Mohagheghi, S., Hernandez, J.C., Harley, R.G.: Particle swarm optimization: Basic concepts, variants and applications in power systems. IEEE Transactions on Evolutionary Computation 12(2), 171–195 (2008)
19. Wright, A.H.: Genetic algorithms for real parameter optimization. In: Rawlins, G.J. (ed.) Foundations of genetic algorithms, pp. 205–218. Morgan Kaufmann, San Mateo (1991)

Guidelines to Select Machine Learning Scheme for Classification of Biomedical Datasets

Ajay Kumar Tanwani, Jamal Afridi, M. Zubair Shafiq, and Muddassar Farooq

Next Generation Intelligent Networks Research Center (nexGIN RC)
National University of Computer & Emerging Sciences (FAST-NU)
Islamabad, Pakistan
{ajay.tanwani,jamal.afridi,zubair.shafiq,muddassar.farooq}
@nexginrc.org

Abstract. Biomedical datasets pose a unique challenge to machine learning and data mining algorithms for classification because of their high dimensionality, multiple classes, noisy data and missing values. This paper provides a comprehensive evaluation of a set of diverse machine learning schemes on a number of biomedical datasets. To this end, we follow a four step evaluation methodology: (1) pre-processing the datasets to remove any redundancy, (2) classification of the datasets using six different machine learning algorithms; Naive Bayes (probabilistic), multi-layer perceptron (neural network), SMO (support vector machine), IBk (instance based learner), J48 (decision tree) and RIPPER (rule-based induction), (3) bagging and boosting each algorithm, and (4) combining the best version of each of the base classifiers to make a team of classifiers with stacking and voting techniques. Using this methodology, we have performed experiments on 31 different biomedical datasets. To the best of our knowledge, this is the first study in which such a diverse set of machine learning algorithms are evaluated on so many biomedical datasets. The important outcome of our extensive study is a set of promising guidelines which will help researchers in choosing the best classification scheme for a particular nature of biomedical dataset.

Keywords: Classification, Machine Learning, Biomedical Datasets.

1 Introduction

Recent advancements in the field of machine learning and data mining have enabled biomedical research to play a direct role in improving the general quality of health care. This fact is supported by a large number of applications developed in the field of *biomedical informatics* to provide solutions to a number of real-world problems. The modern research on mass spectrometry based proteomics, genome-wide association, DNA sequencing and microarrays is made possible by the efficient processing of high-dimensional biomedical data. The trend of keeping permanent medical records in the health management information systems is becoming a standard practice in many countries of the world. Moreover, modern medical equipments and diagnostic techniques generate heterogenous and voluminous data [1]. The ill-structured nature of the biomedical data, thus, require intelligent machine learning and data mining algorithms for automated analysis in order to make logical inferences from the stored raw data.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 128–139, 2009.

A diverse set of machine learning and data mining algorithms have been previously used to extract useful information from the biomedical data. These algorithms usually perform regression, clustering, visualization or classification of the biomedical data in order to assist the medical consultants in the decision making process[1]. The well-known machine learning and data mining based classification algorithms use probabilistic methods, rule-based learners, linear models such as neural networks and support vector machines, decision trees and instance-based learners. Further, a combination of different classification algorithms can result in improved classification accuracy [5]. The commonly used ensemble techniques are *bagging*, *boosting*, *voting* and *stacking*. The use of evolutionary algorithms in recent years is also gaining popularity for discovering knowledge in medical diagnoses [2]. However, their evaluation is beyond the scope of this paper.

Despite the great work and diversity in the existing machine learning schemes, no significant work is done so far to assist a researcher in selecting a suitable classification technique for a particular nature of biomedical dataset. In this paper, we provide a comprehensive empirical study on classification of 31 different biomedical datasets using a diverse set of machine learning schemes. We adopt a four step methodology to ensure a comprehensive evaluation of different machine learning schemes: (1) preprocessing the dataset using attribute selection, (2) providing the preprocessed features' set to six well-known classification algorithms, (3) bagging and boosting each of these classifiers, and (4) creating an ensemble of classifiers by using stacking and voting.

The main subject of this paper is to provide a systematic and unbiased evaluation of the existing machine learning schemes to resolve the uncertainties associated with the choice of classifier and the nature of biomedical data. We follow a question oriented research methodology to resolve a number of pertinent questions like: (1) Can the predictive results of classification be improved by diversity in machine learning schemes or is it largely a function of the dataset under consideration?, (2) What is the significance of the nature of biomedical dataset on classification accuracy?, (3) How various parameters of the dataset (instances, classes, missing values, number of attributes, type of attributes) affect the accuracy of classification?, (4) How the choice of a machine learning scheme affects the classification accuracy?, and (5) Which machine learning schemes are more useful and in what cases? The answers subsequently lead us to propose a number of guidelines that we believe will provide valuable support to researchers working on the classification of biomedical datasets.

Organization of the Paper. In the next section, we provide a brief review of the related work. In Section 3, we discuss the biomedical datasets used in our study. We present a review of our classification schemes in Section 4. In Section 5, we report the results of our experiments which are followed by the standard guidelines. Finally, we conclude the paper with an outlook to our future work in Section 6.

2 Related Work

We now provide a brief overview of recent research done to analyze the accuracy of different machine learning schemes on various biomedical domains. In [3], the authors

[1] The scope of this paper is confined to the classical classification problem for prognosis.

study the impact of feature selection on the classification accuracy using an email and a drug discovery dataset. The authors in [4] present an empirical study of bagging and boosting techniques using neural networks and decision trees on 23 randomly chosen datasets. The results of their study suggest that bagging provides relatively better accuracy compared with each of the individual classifiers whereas boosting produces inconsistent results. The work in [5] evaluates the accuracy of different ensemble combinations of six classification algorithms (LDA, 1-NN, Decision Tree, Logistic Regression, Linear SVMs and MLP) on high-dimensional cancer proteomic datasets. In [6], the authors compare the performance of data mining schemes with the logistic and regression techniques on a clinical database of cancer patients. Their results show that pre-processing the data by attribute selection significantly improves the performance of a classifier while meta-learning is of little value. The study of machine learning methods on Atherosclerosis in [7] involves testing of different categories of machine learning schemes to predict future disorders and death causes. A comprehensive survey of biomedical applications utilizing machine learning schemes is done in [8].

The commonly observed methodology among medical researchers in various papers is to experiment on the dataset with only limited number of algorithms from the machine learning repository and select the one which gives relatively better results for their particular domain. The selection of machine learning algorithms for a particular domain appears to be inclined towards their own view of a particular scheme. Consequently, no guidelines are available to select the best classifier for a particular type of data. In our study, we provide a set of guidelines that will help a researcher in choosing an appropriate classifier based on a particular type of dataset.

3 Biomedical Datasets

Biomedical datasets are generally associated with high-dimensional features and multiple classes. The datasets obtained from clinical databases contain various systemic and human errors [9]. The noisy nature, sparseness and missing values hamper the classification accuracy of the machine learning schemes. These inconsistencies demand to treat the classification problem of biomedical datasets as a separate domain. To comprehensively evaluate the performance of various classification schemes on biomedical datasets, we have selected as many as 31 biomedical datasets publicly available from the UCI Machine Learning repository [10] and Center for Cancer Research [11]. Our selection criterion is to choose well-known datasets from a number of different biomedical domains. The summary of the datasets used in our study is shown in Table 1.

Our repository contains high-dimensional datasets (Ovarian 8-7-02 has a total of $15,154$ attributes), multi-class datasets (Thyroid0387 has a total of 32 classes followed by Cardiac Arrhythmia with 16 classes), imbalanced datasets (class distribution of Hyperthyroid, Cardiac Arrhythmia and Cleveland Heart is highly uneven), datasets with many instances (Protein Data contains $21,618$ instances), datasets with missing values (Hungarian Heart and Horse Colic contains up to 20 percent missing values) and datasets of DNA sequencing and mass spectrometry. We believe that the chosen datasets, therefore, encompass all important domains of biomedicine and bioinformatics.

Table 1. The summary of used datasets: The table shows the name of datasets in the alphabetical order; their year of donation; total number of instances; total classes; number of continuous, binary and nominal attributes; and the percentage of missing values in the attributes

Dataset	Year	Instances	Classes	Attributes			Missing Values (%)
				Continuous	Binary	Nominal	
Ann-Thyroid	1987	7200	3	6	15	0	0
Breast Cancer	1992	699	2	1	0	9	0.23
Breast Cancer Diagnostic	1995	569	2	31	0	0	0
Breast Cancer Prognostic	1995	198	2	33	0	0	0.06
Cardiac Arrhythmia	1998	452	16	272	7	0	0.32
Cleveland-Heart	1990	303	5	10	3	0	0.15
Contraceptive Method	1997	1473	3	2	3	4	0
Dermatology	1998	366	6	1	1	32	0.06
Echocardiogram	1989	132	2	8	2	2	4.67
E-Coli	1996	336	8	7	0	1	0
Haberman's Survival	1999	306	3	3	0	0	0
Hepatitis	1988	155	2	6	0	13	5.67
Horse Colic	1989	368	2	8	4	15	19.39
Hungarian Heart	1991	294	5	10	3	0	20.46
Hyper Thyroid	1989	3772	5	7	21	1	2.17
Hypo-Thyroid	1990	3163	2	7	18	0	6.74
Liver Disorders	1990	345	2	6	0	0	0
Lung Cancer	1992	32	3	0	0	56	0.28
Lymph Nodes	1988	148	4	3	9	6	0
Mammographic Masses	2007	961	2	1	0	4	3.37
New Thyroid	1992	215	3	5	0	0	0
Ovarian 8-7-02	2002	253	2	15154	0	0	0
Pima Indians Diabetes	1990	768	2	8	0	0	0
Post Operative Patient	1993	90	3	0	0	8	0.44
Promoters Genes Sequence	1990	106	2	0	0	58	0
Protein Data	-	21618	3	0	0	1	0
Sick	1989	2800	2	7	21	1	2.24
Statlog Heart	-	270	2	7	3	3	0
Switzerland Heart	1991	123	5	10	3	0	17.07
Thyroid0387	1992	9172	32	7	21	1	5.50
Splice-Junction Gene Sequence	1992	3190	3	0	0	61	0

4 A Review of Classification Schemes

We adopt a four step evaluation methodology to ensure an unbiased evaluation of different machine learning schemes: (1) preprocessing the dataset using attribute selection to remove redundant and useless features, (2) providing the preprocessed features' set to six well-known classification algorithms, (3) bagging and boosting each of these classifiers to analyze their merits in improving the accuracy, and (4) finally creating a team of classifiers by combining the the best version (individual, bagged and boosted) of each of the six base classifiers using stacking and voting in order to further enhance the accuracy. We use the standard implementations of these schemes in Wakaito Environment for Knowledge Acquisition (WEKA) [12].

4.1 Data Pre-processing

The attribute selection technique [13] is used as a pre-processing filter to remove the redundant or useless features in the dataset. We use Best First algorithm for the attribute selection that performs greedy hill climbing with a backtracking search method [12].

4.2 Base Classifiers

Naive Bayes. Naive Bayes (NB) utilizes a probabilistic method for classification by multiplying the individual probabilities of every attribute-value pair [14]. This simple algorithm assumes independence among the attributes and even then provides excellent classification results.

Neural Networks using Multi Layer Perceptron. The Multi Layer Perceptron (MLP) consists of input layer (attributes), output layer (classes) and hidden layer(s) that are interconnected through various neurons. The back propagation algorithm tends to optimize the weights of these connections through training instances of the dataset [15]. We have used default parameters for MLP in WEKA. The number of epochs is equal to 500, the learning rate is 0.3 and the momentum of updating weights is 0.2.

Support Vector Machines using Sequential Minimal Optimization. The Support Vector Machine (SVM) algorithm builds a hyperplane to separate different instances into their respective classes [18]. A pairwise classification scheme is used to do multi-class classification. We use Sequential Minimal Optimization (SMO) which is a fast and an efficient version of SVM implemented in WEKA.

Instance Based Learner. The Instance Based Classifier (IBk) is the simplest among the algorithms used in our study [16]. The classification is done on the basis of a majority vote of k neighboring instances. We have used k=5 while taking default values of WEKA for rest of the parameters. The window size is zero that allows maximum number of instances in the training pool without replacements.

Decision Tree. The decision tree (J48) is an implementation of C4.5 in WEKA. The tree comprises of nodes (attributes) at every stage that are structured with the help of training examples [17].

Inductive Rule Learner. Repeated Incremental Pruning to Produce Error Reduction (RIPPER) is a propositional rule learner that defines a rule based detection model and seeks to improve it iteratively by using different heuristic techniques [19]. The constructed rule set is then used to classify new instances. We have implemented this rule based system in WEKA using JRIP with default parameters.

4.3 Resampling Based Ensembles

Bagging. Bagging combines the multiple models generated by training a single algorithm on random sub-samples of a given dataset [20]. Unbiased voting is used during the fusion process.

Boosting. Boosting, in contrast to bagging, uses weighted voting to generate more misclassified instances in its successive models [21].

4.4 Meta-learning Based Ensembles

Stacking. Stacking combines the outputs of two or more base-level classifiers by training them with a meta-learner [22]. In all of our experiments, we use Naive Bayes as

a standard meta-learner while the ensemble comprises of the best version (individual, bagging and boosting) of each of the six different base classifiers.

Voting. Voting is a meta-learning technique that uses different combinations of probability estimates of base classifiers for classification [23]. The selection criteria for choosing the six base learners for voting is same as that of stacking. We have implemented voting in WEKA using an average of the probabilities as the combination rule.

5 Experiments, Results and Guidelines

We now report the results of our experiments that we have done to analyze the classification accuracy of 20 different types of machine learning algorithms on 31 different biomedical datasets. We use Area Under an ROC (Receiver Operating Characteristic) Curve (AUC) ($0 \leq$ AUC ≤ 1) metric to quantify the classification accuracy of an algorithm. AUC is known to be a 'more complete' performance metric as compared to other traditionally used metrics [24], [25]. The ROC curves are generated by varying the threshold on output class probability. AUC = 100% represents the best accuracy while AUC = 0% represents the worst accuracy. The results in Table 2 show the mean AUCs of the machine learning algorithms used in our comparative study on biomedical datasets. Some of the experiments could not be completed even after running for several days and are indicated by blank spaces in our results. We now present our analysis and important insights on the basis of the results obtained from these experiments. Our primary motivation is to to investigate the factors that can potentially affect the classification accuracy of a particular machine learning scheme. The main variables that determine the classification accuracy are categorized by: (1) the nature of a dataset, (2) the pre-processing filter, and (3) the choice of a classification scheme.

5.1 How Does the Nature of a Dataset Affect the Classification Accuracy?

The classification accuracy of a given algorithm is largely dependent on the nature of dataset rather than the algorithm itself. The main characteristics of a dataset are its attributes, classes and number of instances. We answer the following pertinent questions to systematically study the nature of a dataset.

Role of Attributes. The attributes of a dataset vary in terms of their quality, number and type (continuous, binary or nominal). The quality of information that attributes can provide is an important factor that determines the *classification potential* of a dataset. The quality of information can be quantified using well-known parameters like information gain, entropy, gain ratio etc. We use information gain in our study. The results of our experiments demonstrate that the classification accuracy is directly proportional to the information gain of a dataset. The low information gain of datasets like Protein Data, Liver Disorders and Haberman's Survival etc is mainly responsible for relatively poor classification accuracy of all algorithms on them. For example, the Protein Data dataset has only one attribute with an information gain of just 0.0647 which results in the best mean AUC value of only 63.27% among all the applied machine learning schemes.

Table 2. Mean AUCs for biomedical datasets using a diverse set of machine learning algorithms. The bold entries in every row represent the best results. The blank spaces are used to show the missing results. The acronym (Bag) is used for *bagging* while (Bos) represents *boosting*.

Dataset	Individual Classifiers																		Team of Classifiers	
	NB			MLP			SMO			IB*k*			J48			JRIP				
	NB	NB (Bos)	NB (Bag)	MLP	MLP (Bos)	MLP (Bag)	SMO	SMO (Bos)	SMO (Bag)	IB*k*	IB*k* (Bos)	IB*k* (Bag)	J48	J48 (Bos)	J48 (Bag)	JRIP	JRIP (Bos)	JRIP (Bag)	Stacking	Voting
Ann-Thyroid	96.52	86.24	96.31	99.08	91.29	99.13	62.21	86.81	65.01	94.26	90.11	95.98	99.12	99.46	**99.59**	98.99	99.44	99.55	99.28	99.49
Breast Cancer	98.53	98.35	98.80	98.65	97.56	99.00	96.29	96.88	96.96	99.11	97.95	**99.16**	95.85	98.74	98.74	96.51	98.26	98.69	97.62	99.19
Breast Cancer Diagnostic	98.30	98.20	98.70	98.30	97.50	**99.20**	95.60	98.30	96.70	98.70	96.90	98.80	94.20	98.90	98.40	94.60	99.00	98.50	98.20	99.10
Breast Cancer Prognostic	73.20	70.20	**73.70**	70.60	64.30	72.90	50.00	66.60	49.70	68.00	64.10	67.40	53.80	69.00	66.20	59.80	65.70	69.50	70.10	73.40
Cardiac Arrhythmia	66.36	70.98	66.75	70.19	72.89	77.38	65.52	53.57	67.15	73.91	69.86	72.56	66.13	84.92	75.40	64.36	83.27	73.45	66.22	**94.78**
Cleveland-Heart	76.40	60.18	76.30	72.10	75.64	74.00	69.88	66.66	70.90	71.34	61.22	71.80	57.22	73.20	72.36	55.66	57.52	66.34	**79.02**	77.26
Contraceptive Method	70.57	65.50	70.20	72.70	68.23	**74.00**	63.40	62.70	65.17	68.93	65.50	69.60	71.20	68.47	71.07	64.17	64.17	68.30	73.17	73.10
Dermatology	**99.75**	99.20	99.70	99.15	98.82	99.63	98.98	99.73	99.62	99.43	97.58	99.50	97.35	98.98	99.07	97.30	99.33	99.23	-	99.82
Echocardiogram	82.75	83.20	83.05	81.70	82.75	79.85	79.70	80.90	79.65	80.20	77.80	81.35	81.45	81.65	82.60	78.05	83.30	79.60	80.45	**83.50**
E-Coli	74.38	75.84	-	82.29	82.29	-	85.99	86.34	-	87.18	77.93	-	77.90	83.22	-	76.89	**91.93**	-	-	-
Haberman's Survival	**69.70**	66.80	68.00	68.60	58.50	68.30	50.00	65.50	52.40	63.50	56.90	65.90	53.10	63.40	66.80	60.40	65.00	64.90	67.30	67.50
Hepatitis	**89.10**	83.74	89.05	85.77	81.87	86.51	67.47	87.37	79.41	81.96	72.05	84.68	65.60	82.16	83.22	67.20	67.20	78.76	88.35	88.64
Horse Colic	86.30	84.10	86.40	85.10	84.30	87.30	81.40	86.40	82.40	86.00	82.60	86.50	86.80	85.90	88.30	83.50	87.20	88.50	89.20	**89.90**
Hungarian Heart	89.90	87.20	90.20	87.60	85.80	89.80	78.40	88.40	81.00	79.30	64.20	84.60	78.90	87.90	87.10	75.00	88.00	86.80	89.95	**90.70**
Hyper Thyroid	92.28	93.30	92.46	90.80	86.70	**95.98**	62.20	92.68	55.96	88.84	70.84	83.12	78.58	94.80	85.26	69.42	93.94	77.70	81.42	96.64
Hypo-Thyroid	97.58	96.91	97.58	98.55	94.20	98.84	64.14	97.82	68.06	88.18	87.46	89.39	95.45	98.14	97.20	95.05	97.29	95.47	98.14	**98.90**
Liver Disorders	57.90	58.70	57.00	59.70	59.00	**61.00**	50.00	58.60	50.00	56.50	56.70	57.50	55.60	55.80	58.50	56.00	59.80	57.20	59.70	60.50
Lung Cancer	95.65	82.37	95.17	90.82	90.82	90.82	73.43	88.89	81.88	93.72	91.54	90.82	72.95	81.64	85.51	74.88	89.13	76.33	86.36	**96.23**
Lymph Nodes	87.40	71.89	88.30	92.70	83.01	95.56	92.70	82.04	92.99	93.94	75.21	**96.25**	74.85	87.15	93.09	71.96	92.70	90.64	71.23	96.23
Mammographic Masses	89.60	86.40	89.50	87.00	85.70	89.60	79.60	85.00	84.70	85.00	82.00	86.30	85.80	86.80	88.80	84.70	89.00	88.60	89.70	**89.80**
New Thyroid	99.55	99.67	99.63	99.70	98.80	99.67	89.37	99.43	93.63	97.93	97.20	97.90	91.00	99.50	98.00	91.87	97.47	99.27	99.03	**99.77**
Ovarian 8-7-02	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	97.10	97.12	98.77	94.80	97.52	98.77	**100**	-
Pima Indians Diabetes	**82.90**	79.60	82.80	80.50	79.20	82.50	71.50	75.60	74.10	77.90	69.50	79.40	79.10	77.90	80.90	72.00	78.10	77.70	82.45	82.80
Post Operative Patient	31.53	45.50	31.96	45.20	**55.50**	44.07	48.93	39.90	47.20	34.50	42.43	36.23	33.17	36.67	32.70	33.53	33.20	32.60	29.40	44.40
Promoters Genes Sequence	93.40	95.39	**97.90**	**97.90**	**97.90**	97.80	95.30	95.28	95.20	96.60	94.87	97.20	87.70	97.69	93.10	83.00	96.01	97.70	95.90	-
Protein Data	63.17	54.47	63.27	61.83	56.70	63.10	54.00	54.47	55.23	63.17	54.47	**63.27**	50.00	50.00	50.00	50.00	50.00	50.00	-	62.80
Sick	93.09	88.55	93.21	**94.88**	90.83	94.38	49.98	91.42	54.54	89.40	86.55	86.82	90.05	92.48	91.22	89.20	92.57	91.26	92.83	94.57
Statlog Heart	89.15	86.10	89.38	84.17	87.54	87.83	84.42	84.62	86.55	85.36	81.49	86.18	80.16	85.08	89.08	78.88	85.02	88.60	88.49	**90.37**
Switzerland Heart	47.44	47.44	47.88	53.72	53.72	53.96	53.94	53.94	48.04	46.76	47.14	47.42	47.30	47.30	47.56	41.64	41.64	47.66	**59.44**	49.12
Thyroid0387	**89.60**	-	-	79.86	-	-	73.87	-	-	75.91	-	-	85.21	-	-	75.09	-	-	-	-
Splice-Junction Gene Sequence	**98.78**	96.49	**98.78**	98.01	96.03	98.56	95.94	96.94	97.49	96.85	93.02	96.99	93.98	98.03	98.02	95.48	97.75	97.94	98.25	98.77
Mean	83.34	80.57	83.52	83.62	81.90	84.85	73.25	80.57	74.88	81.35	76.79	81.78	76.32	82.03	81.95	75.1	80.98	80.68	82.63	**85.08**

Table 3. Classification differences with and without attribute selection as pre-processor. Bold entries in every row represent the best accuracy.

Dataset	With Pre-Processing			Without Pre-Processing		
	Total Attributes	Net Information Gain	Best Mean AUC	Total Attributes	Net Information Gain	Best Mean AUC
Liver Disorder	1	0.051	61.00	6	0.057	**75.40**
Haberman's Survival	1	0.072	69.70	3	0.072	**71.20**

Guideline 1: Use information gain to quantify the quality of attributes in order to determine the classification potential of a dataset.

Role of output Classes. The multiple output classes lead to imbalanced datasets when the class distribution is not even. Our experiments reveal that the multiclass imbalanced datasets pose a significant challenge in terms of the classification accuracy. For example, the class distributions of Cardiac Arrhythmia (16 classes) and Cleveland Heart (5 classes) datasets are highly imbalanced in favor of one class that logically results in their relatively low mean AUC values. However, the accuracy significantly improves if we deploy a team of classifiers. For example, in case of Cardiac Arrhythmia dataset, the classification accuracy improves from best mean AUC value of 84.92% obtained with all the individual classifiers to 94.78% when the meta-learning technique of voting is used. In comparison, for Cleveland Heart dataset, the best mean AUC value increases from 76.4% to 79.02% when stacking is used.

Guideline 2: Use a team of classifiers for multi-class imbalanced datasets.

Role of Instances. The number of instances, however, have little role on the classification accuracy of algorithms. It is the quality of instances quantified with the help of information gain, which determines the classification potential of a dataset. For example, the Lung Cancer dataset has only 32 instances compared to $21,618$ instances of Protein Data dataset. However, the best mean AUC for the former is 95.95% while for the later it is just 63.27%. The information gain for both the datasets are respectively 1.521 and 0.0647. This proves our thesis that the large AUC for Lung Cancer dataset even with small instances is due to the large information gain of its attributes.

Guideline 3: Do not contemplate on the classification potential of a dataset on the basis of its number of instances only.

5.2 When to Use the Pre-processing Filter?

The attribute selection is used as a pre-processer to remove the redundant and useless attributes in a dataset. The pre-processing filter in most of the cases improves the classification accuracy of datasets with the exception of few ones. Therefore, it is important to identify when to use a pre-processing filter. Our study again suggests that the decision should be based on the information gain of attributes. If the net information gain of a dataset is small or the number of attributes become too less after pre-processing, then the pre-processing filter should not be used. In Table 3, we report the results with and without pre-processing filter on two of the datasets (Liver Disorder and Haberman's Survival) which are relatively challenging for classification. The results prove our hypothesis that we should not use a pre-processing filter if it further degrades the quality of attributes.

Table 4. Mean AUCs and standard deviations of six base classifiers over all datasets used in this study. Bold entries in every row represent the best accuracy.

Classification Scheme	NB	MLP	SMO	IB*k*	J48	JRIP	Mean
Individual	83.34 ± 16.99	**83.62 ± 15.19**	73.25 ± 17.15	81.35 ± 16.54	76.32 ± 17.68	75.1 ± 17.47	78.83 ± 16.84
Bagging	83.52 ± 16.05	**84.85 ± 14.57**	74.88 ± 16.89	81.78 ± 16.32	81.95 ± 17.18	80.67± 18.73	81.28 ± 16.62
Boosting	80.57 ± 22.88	81.90 ± 21.67	80.57 ± 22.26	76.79 ± 22.05	**82.03 ± 22.83**	80.98 ± 22.95	80.48± 22.44
Mean	82.48 ± 18.64	**83.46 ± 17.14**	76.23 ± 18.77	79.97 ± 18.31	80.10 ± 19.23	78.92 ± 19.72	80.20± 18.63

Guideline 4: Do not use attribute selection as a pre-processor filter on the datasets if : (1) they have low quality information attributes, or (2) the remaining attributes after the preprocessing are too less to be of any value.

5.3 How Does the Machine Learning Scheme Affect the Classification Accuracy?

In this section, we analyze the effect of different machine learning algorithms on classification accuracy of a dataset.

Resampling based Ensembles vs Individual Classifiers? Resampling Based Ensemble techniques are preferable over individual classifiers because the final classification is done by training the algorithm on different regions of the sample space. As a result, these ensembles reduce the over fitting bias of an algorithm. Table 4 provides the net mean AUC's of six base classifiers with resampling based ensembles over all the datasets. It is clear that combining multiple resampling methods for classifier enhancement (such as bagging or boosting) are generally more effective than the individual classifier. Moreover, it is only 30 out of 174 times (17.24%) when a single classifier produced better accuracies than the respective bagging and boosting models of the classifiers. Our results show that the overall mean AUC of resampling based ensembles is 80.86% compared to that of 78.83% for individual classifiers.
Guideline 5: Use resampling based classifier enhancement techniques (bagging and boosting) over individual classifiers.

When is Bagging particularly useful? Bagging neutralizes the instability of algorithms by using unbiased voting procedure for combining multiple samples [12]. This explains the reason behind the better average AUCs of bagging for all the individual classifiers. We can see in Table 4 that average AUC for bagging is 81.28% compared with 80.48% of boosting and 78.83% of the individual classifiers. Moreover, our results show that 130 out of 179 times (72.63 %) bagging has improved the accuracy of individual classifiers. These insights support our argument bagging is a particularly useful technique for classifier enhancement.
Guideline 6: Use bagging as classifier enhancement to improve the classification accuracy of the individual algorithms.

When is Boosting particularly useful? The biased voting and weighted selection of instances in boosting often gives inconsistent results compared with those of bagging or individual classification schemes. The reason is that boosting over fits on noisy datasets

[4]. The unpredictable behavior of boosting often leads to significantly low AUCs for unstable algorithms. For example, boosted IB*k* in Hepatitis dataset decreases the mean AUC value of individual IB*k* from 81.96% to 72.05%. Similarly, for the Hyperthyroid dataset, the mean AUC value of boosted IB*k* is 70.84% compared with AUC value of 88.84% for the individual classifier. In comparison, boosting significantly improves the AUC values for stable algorithms. For example, boosted SMO in Sick dataset increases the AUC of SMO from 49.98% to 91.42%; boosted JRIP in Cardiac Arrhythmia dataset increases the AUC of JRIP from 64.36% to 83.27%; and boosted J48 in Hyperthyroid dataset increases the mean AUC value of J48 from 78.58% to 94.80%. We can see in Table 2 that the improvements due to boosting on SMO, JRIP and J48 are scalable to other datasets as well. Boosting is particulary suited for SMO because its average AUC values are 80.57% compared with 74.88% of bagged SMO and simple SMO of 73.25%.
***Guideline 7: Use boosting on stable algorithms like SMO, JRIP, and J48 and do not use it on unstable algorithms like MLP and IB*k.**

Bagging Naive Bayes vs Individual Naive Bayes? Naive Bayes results are excellent for datasets like Haberman, Hepatitis, Ovarian 8-7-02, Pima Indian Diabetes and Splice Junction Gene Sequencing. The classification accuracy of bagging and simple Naive Bayes is in general better than the boosted Naive Bayes. Therefore, it becomes relevant to have a guideline when to enhance Naive Bayes with bagging? The problem can be analyzed by dividing the *significant attributes* (the attributes after the attribute selection phase) in two groups: (1) continuous and multinominal attributes having more than *n* values, and (2) multinominal attributes having less than *n* values. If the net information gain of the first group is greater than that of the second group, then use bagging Naive Bayes. This conjecture works well with $n = 4$. For example, the significant attributes in Hyperthyroid dataset comprise of 3 continuous and 2 binary valued attributes and the information gain distribution is: (1) *total information gain of multinominal attributes with less than 4 values = 0.0335*, and (2) *total information gain of other remaining attributes = 0.14*. The results in Table 2 show that the classification accuracy increases from 92.28% to 92.46% in favor of bagging Naive Bayes compared to the individual Naive Bayes. In a similar way, the information gain distribution of Cleveland Heart dataset after attribute selection is: (1) *total information gain of multinominal attributes with less than 4 values = 0.847*, and (2) *total information gain of other remaining attributes = 0.347*. It can be seen in Table 2 that individual Naive Bayes proved to be better in this case with a mean AUC of 76.4% compared to 76.3% obtained from bagging Naive Bayes. The datasets like Breast Cancer Prognostic, Breast Cancer Diagnostic, Lung Cancer, Contraceptive Method etc. all support this conjecture.
Guideline 8: Use bagged version of Naive Bayes instead of individual one only if after attribute selection, the net information gain of continuous and multinominal attributes with more than* n *values (*$\mathbf{n = 4}$*) is greater than the information gain of multinominal attributes with less than* n *values.

Meta-Learning Based Ensembles - Voting vs Stacking? The criterion that we use to select the base classifiers for making a good team of classifiers is based on both their diversity and accuracy. We choose the best version among individual classifier, their bagged and boosted version for each of the six different individual classifiers, and

use Naive Bayes as a meta-learner to produce a meta-learning ensemble. Our experiments demonstrate that stacking in general do not improve the classification accuracy of medical datasets. The mean AUC values of stacking are comparable with those of other techniques. On the other hand, the classification accuracy of voting is much better than those of all other classification techniques with an overall average AUC value of 85.08%.

Guideline 9: Use voting instead of stacking for meta-learning ensembles to achieve better AUC values.

Which classification algorithm is the best? We choose the best classification algorithm on two parameters: (1) overall classification accuracy, and (2) variance in accuracy that determines the stability and consistency of an algorithm. We can see from Table 4 that *Bagging MLP not only gives on the average the best overall classification accuracy with an AUC value of* 84.85% *but also the least standard deviation of* 14.57.

Guideline 10: Use bagging MLP for classification if the nature of a biomedical dataset is unknown.

6 Conclusion

In this paper, we have presented a comprehensive empirical study of a diverse set of machine learning algorithms on a large number of biomedical datasets. The diversity is added by using resampling based ensemble methods of bagging and boosting and meta-learning techniques of stacking and voting. We conclude that the nature of a given dataset plays an important role on the classification accuracy of algorithms; therefore, it is imperative to choose an appropriate algorithm for a particular dataset. We have identified some general characteristics of a dataset that can be useful in selecting the most suitable algorithm as per the nature of underlying dataset. We have also evaluated the performance of various machine learning schemes under different scenarios to study the effect of diversity on the classification results. The results of our experiments show that voting in general is the most powerful technique among the compared machine learning schemes. On the basis of our study, we have been able to formulate 10 generic guidelines that can help researchers of biomedical classification community to select an appropriate classifier for their particular problem. In future, we would like to devise a metaheuristic framework that can recommend the most suitable classifier for the dataset by analyzing the patterns in the dataset.

Acknowledgements. The authors of this paper are supported, in part, by the National ICT R&D Fund, Ministry of Information Technology, Government of Pakistan. The information, data, comments, and views detailed herein may not necessarily reflect the endorsements of views of the National ICT R&D Fund.

References

1. Wasan, S., Bhatnagar, V., Kaur, H.: The impact of data mining techniques on medical diagnostics. Data Science Journal 5, 119–126 (2006)
2. Pena-Reyes, C.A., Sipper, M.: Evolutionary computation in medicine: an overview. Journal of Artificial Intelligence in Medicine 19(1), 1–23 (2000)

3. Janecek, A.G.K., Gansterer, W.N., Demel, M.A., Ecker, G.F.: On the relationship between feature selection and classification accuracy. Journal of Machine Learning and Research 4, 90–105 (2008)
4. Opitz, D., Maclin, R.: Popular ensemble methods: an empirical study. Journal of Artificial Intelligence Research 11, 169–198 (1999)
5. Assareh, A., Moradi, M.H., Volkert, L.G.: A hybrid random subspace classifier fusion approach for protein mass spectra classification. In: Marchiori, E., Moore, J.H. (eds.) EvoBIO 2008. LNCS, vol. 4973, pp. 1–11. Springer, Heidelberg (2008)
6. Hayward, J., Alvarez, S., Ruiz, C., Sullivan, M., Tseng, J., Whalen, G.: Knowledge discovery in clinical performance of cancer patients. In: IEEE International Conference on Bioinformatics and Biomedicine, USA, pp. 51–58 (2008)
7. Serrano, J.I., Tomeckova, M., Zvarova, J.: Machine learning methods for knowledge discovery in medical data on Atherosclerosis. European Journal for Biomedical Informatics 2(1), 6–33 (2006)
8. Kononenko, I.: Machine learning for medical diagnosis: History, state of the art and perspective. Artificial Intelligence in Medicine 23(1), 89–109 (1995)
9. Lavrac, N.: Selected techniques for data mining in medicine. Artificial Intelligence in Medicine 16, 3–23 (1999)
10. UCI repository of machine learning databases, University of California-Irvine, Department of Information and Computer Science, www.ics.uci.edu/~mlearn/MLRepository.html
11. Ovarian cancer studies, center for cancer research, National Cancer Institute, USA, http://home.ccr.cancer.gov/ncifdaproteomics/ppatterns.asp
12. Witten, I.H., Frank, E.: Data mining: practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
13. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning and Research 3, 1157–1182 (2003)
14. Rish, I.: An empirical study of the naive Bayes classifier. In: IJCAI Workshop on Empirical Methods in Artifical Intelligence, pp. 41–46 (2001)
15. Haykin, S.: Neural networks: a comprehensive foundation, 2nd edn. Pearson Education, London (1998)
16. Aha, D.W., Kibler, D., Albert, M.K.: Instance based learning algorithms. Machine Learning 6(1), 37–66 (1991)
17. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann, San Francisco (1993)
18. Vapnik, V.N.: Statistical learning theory. Wiley Interscience, USA (1998)
19. Cohen, W.W.: Fast effective rule induction. In: Proceedings of Twelfth International Conference on Machine Learning, USA, pp. 115–123 (1995)
20. Breiman, L.: Bagging Predictors. Machine Learning 24(2), 123–140 (1996)
21. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: Proceedings of the Thirteenth International Conference on Machine Learning, Italy, pp. 148–156 (1996)
22. Ting, K.M., Witten, I.H.: Stacked generalization: when does it work. In: Proceedings of the Fifteenth IJCAI, pp. 866–871. Morgan Kaufmann, San Francisco (1997)
23. Abe, H., Yamaguchi, T.: Constructive meta-learning with machine learning method repository. In: Orchard, B., Yang, C., Ali, M. (eds.) IEA/AIE 2004. LNCS, vol. 3029, pp. 502–511. Springer, Heidelberg (2004)
24. Fawcett, T.: ROC graphs: notes and practical considerations for researchers, TR HPL-2003-4, HP Labs, USA (2004)
25. Walter, S.D.: The partial area under the summary ROC curve. Statistics in Medicine 24(13), 2025–2040 (2005)

Evolutionary Approaches for Strain Optimization Using Dynamic Models under a Metabolic Engineering Perspective

Pedro Evangelista[1,2], Isabel Rocha[2], Eugénio C. Ferreira[2], and Miguel Rocha[1]

[1] Departament of Informatics / CCTC - University of Minho
Campus de Gualtar, 4710-057 Braga - Portugal
ptiago@deb.uminho.pt mrocha@di.uminho.pt
[2] IBB - Institute for Biotechnology and Bioengineering
Center of Biological Engineering - University of Minho
Campus de Gualtar, 4710-057 Braga - Portugal
{ecferreira,irocha}@deb.uminho.pt

Abstract. One of the purposes of Systems Biology is the quantitative modeling of biochemical networks. In this effort, the use of dynamical mathematical models provides for powerful tools in the prediction of the phenotypical behavior of microorganisms under distinct environmental conditions or subject to genetic modifications.

The purpose of the present study is to explore a computational environment where dynamical models are used to support simulation and optimization tasks. These will be used to study the effects of two distinct types of modifications over metabolic models: deleting a few reactions (knockouts) and changing the values of reaction kinetic parameters. In the former case, we aim to reach an optimal knockout set, under a defined objective function. In the latter, the same objective function is used, but the aim is to optimize the values of certain enzymatic kinetic coefficients. In both cases, we seek for the best model modifications that might lead to a desired impact on the concentration of chemical species in a metabolic pathway. This concept was tested by trying to maximize the production of dihydroxyacetone phosphate, using Evolutionary Computation approaches. As a case study, the central carbon metabolism of *Escherichia coli* is considered. A dynamical model based on ordinary differential equations is used to perform the simulations. The results validate the main features of the approach.

1 Introduction

Systems Biology represents a new approach to research in Biology. It aims to achieve the understanding of the complex interactions in biological systems under an integrative approach, where the ultimate goal is to simulate these systems under different scenarios and perturbations [20]. One of the main purposes of this work is to provide tools for the dynamical modeling and optimization of biological processes, under a Metabolic Engineering perspective. Indeed, we aim

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 140–151, 2009.

to provide tools to identify optimal or near-optimal sets of genetic changes in microorganisms under dynamical conditions to achieve a given industrial aim.

Mathematical dynamical models allow to study the interaction of biological compounds in cells. There are several types of dynamic models [13] [2], but the most common approach is to represent metabolic networks as a system of ordinary differential equations (ODEs). One of the major drawbacks with these types of models is how to reliably estimate model parameters. Another pertinent question is how these values can change and the biological meaning of those modifications. In this work, we face the question of how to evolve a dynamical model based on predefined goals.

We take a new approach by using dynamical models and Evolutionary Computation to identify a set of knockouts or variations in some kinetic parameters that will optimize the production of a certain metabolite. Each configuration is evaluated resorting to a simulation using the dynamical model. This type of information can help to assess on how to engineer a metabolic network in order to enhance the production of a given metabolite and can also be used to infer regulatory data.

It is important to bear in mind that finding a knockout set can be seen as a change in the model structure and the corresponding problem belongs to the class of combinatorial optimization. On the other hand, the second task involves finding the best values for a number of parameters, thus a numerical optimization task. Thus, although the two tasks are quite different from the point of view of optimization, a similar and general purpose strategy will be followed in both cases.

Indeed, to study the described scenarios the concept of dynamical model evolution is introduced. In our proposal, a model will evolve based on a fitness function that is defined considering a given industrial aim (e.g. to maximize the concentration of a given metabolite along the time of the experiment). In alternative, although this is not shown in this work, the fitness can also be based on an error function, if some experimental data to fit is available.

An optimization framework was built around this concept, where the major design concern was the loose coupling between the optimization and the simulation modules. This allows us to optimize any model component independently of the optimization algorithm and of the simulation method.

The framework was applied in this work to the dynamical model of the central carbon metabolism of *Escherichia coli* [2]. This model links the sugar transport system with the reactions of glycolysis and pentose-phosphate pathway. The case study was chosen because it includes most of the reactions of the central carbon metabolism and has been validated experimentally. Moreover, *E. coli* has been the organism of choice to test novel Metabolic Engineering tools, given the simplicity in performing genetic modifications, among other factors.

The optimization of knockout sets to enhance the production of metabolites has been approached before in literature [14] [16]. These studies focused in finding a knockout set using stoichiometric models, performing the simulations using steady-state approaches such as Flux Balance Analysis [4]. Rather less

attention has been paid to optimizing a knockout set based on dynamical models, using as a fitness function the concentration of a certain metabolite in a defined time interval.

Several methods have also been proposed to estimate parameter values based on experimental data [12]: Wang [19] applied an extra focus in how to use genetic algorithms to optimize model parameters. In [10], the authors describe the use of Evolutionary Algorithms (EAs) to reconstruct a metabolic network using functional Petri Nets. In the work developed by Haunschild [6], the concept of automatic generation/evolution of multiple metabolic models is introduced to try to explain some biochemical network phenomena. However, rate equations are defined by the user and are not evolved themselves. These approaches are orthogonal to the one presented in this paper, since in our problem context there is no need of using experimental data to find parameter values.

The rest of this paper is organized as follows: the description of our framework for dynamical model evolution is given in section 2; in section 3, the case study is presented; afterwards, in section 4 the results are presented and discussed; finally, section 5 provides the conclusions and the future work.

2 A Framework for Dynamical Model Evolution

This section describes our general framework for dynamical model evolution. As said before, a dynamical model can evolve based on a given fitness function that can be defined in a flexible way, i.e. no restrictions are imposed over the definition of the fitness function (it can be nonlinear, non differentiable, discontinuous, etc.). It can, for instance, be an error function that takes into account known experimental data and thus the aim will be to estimate the parameters that best fit the data. On the other hand, the fitness function can be based on the concentration of one or several metabolites, along the simulation period. In this last case, the fitness can be measured by the integral (area under the curve) of the objective function.

Our framework is divided into two logical parts: model simulation and optimization (Figure 1). The simulation part is based on the numerical integration of the ODEs of the model, specifying a time interval and considering a fixed model structure with pre-defined values for the model parameters. A set of initial values (e.g. representing environmental constraints) can also be defined by the user for the state variables of the model.

The optimization part allows modifications both in the model structure and in several types of parameters, including kinetic formulas and corresponding parameters. The purpose is to reach model configurations that optimize a given fitness function. A user can impose changes over the model in order to simulate specific cases. Furthermore, optimization algorithms can be defined to search over the space of potential solutions, given the type of allowed changes, that can be summarized in the following:

– Changes in the initial values of the variables (e.g. initial metabolite concentrations);

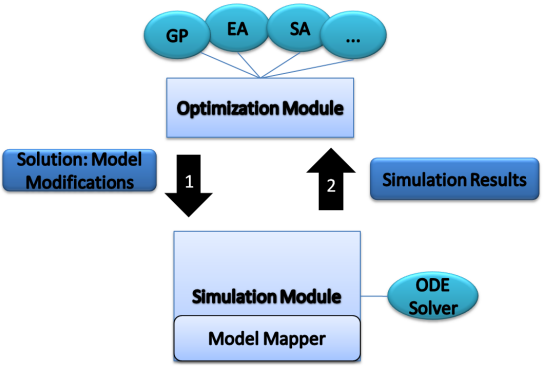


Fig. 1. Framework for dynamical model evolution

- Changes in the kinetic parameters (e.g. global model parameter or parameters of a specific kinetic expression);
- Changes in the structure of the reaction kinetics (e.g. algebraic expression);
- Changes in the overall model structure (e.g. reaction participant metabolites).

In this paper, the main focus will be on the kinetic parameter variation and model structure, considering the possible deletion of a number of reactions from the model.

In our framework, an optimization process is represented by a model, an optimization algorithm and corresponding parameters, a decoder and an override model, while a simulation is only characterized by a model and the override model component. The algorithm and the parameters represent the optimization method and the variables that will be used during a optimization run.

The model integrates all components that describe the dynamical system (the ODEs, the kinetic laws and parameters, etc.). For this purpose, a unified model representation is built, called *model mapper*, that will answer any queries about the model components (e.g. about the model structure or parameter values). This model view is composed by three layers in the following order (Figure 2): (3) the original model, (2) the decoder and (1) the override model. When a query is made it is passed along the chain of entities (in the order 1,2,3) until one of them can answer the query.

Therefore, the decoder and the override model are fractional model representations. The decoder gives a partial model view based on a specific encoding. This layer is used mainly to provide a way to decode the solutions of possible optimization algorithms from their internal representations, namely decoding the genome of an EA. The override model can be used to redefine a set of model components, thus enabling to set conditions that remain constant throughout the optimization process.

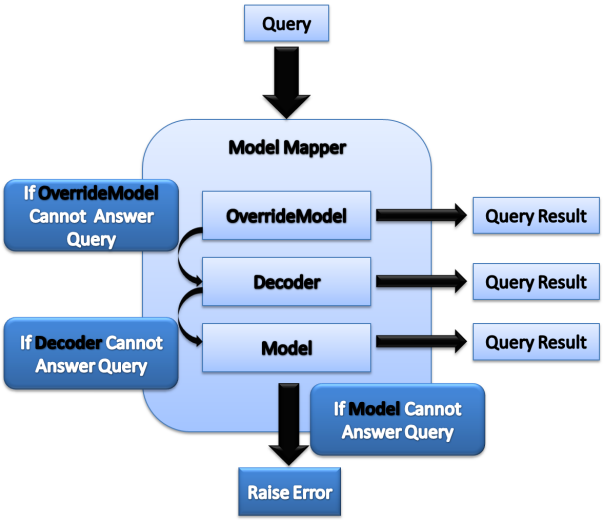


Fig. 2. Framework layers for dynamical model evolution

In more detail, a *model* is composed by the following elements:

- A set of parameters. Each parameter is denoted by a name and has a numerical value.
- A set of variables. A variable is defined by an upper and a lower limit, an initial value and an ODE, that is represented by a sum of terms, where each term has a multiplicative coefficient and a function.
- A set of functions. A function can be any mathematical entity that receives as its parameters the current time and a model representation, returning a numerical result. Functions can also have a local parameter space that overrides the model global parameter scope.

Regarding the optimization layer, several algorithms can be employed, providing they are able to deal with the type of fitness functions described before. Given the complexity of the underlying problems, the available options are meta-heuristics that range from Multistart Local Search to Simulated Annealing, contemplating also several evolutionary approches, such as EAs, Genetic Programming or Differential Evolution (DE). In this work, EAs will be used to perform combinatorial optimization and a DE will perform the numerical optimization task. The specific features of these algorithms will be presented in the next section.

Besides running the simulation and optimization of dynamical models, the framework also allows to input models using the Systems Biology Markup Language (SBML) [7] format (a standard in these kind of models). The results of a simulation or optimization process can also be saved in a text file. A number of visualization tools are also available to allow the user to perform a graphical analysis over the outputs.

Regarding its implementation, the software for the proposed framework was developed using the Java programming language and the following additional libraries: a library for EAs developed by the authors, *CVODE* [3] using JNI that solves systems of ODEs, *JFreeChart* [9] that displays graphical simulation results and *LibSBML* [1] that parses SBML encoded files.

3 Case Study

3.1 Dynamical Model of the Central Carbon Metabolism of *E. coli*

In this paper, a case study on the dynamical model of glycolysis and the pentose-phoshate pathway in *Escherichia coli*[2] was used. One of the main ambitions in Metabolic Engineering is the re-engineering of biological pathways with the aid of mathematical models. The model was delineated and corroborated based in metabolite concentration measurements obtained at transient conditions. This model allows to explore this network as supplier of precursors. For example, dihydroxyacetone phosphate (DHAP) can be produced and used in the lipid synthesis pathway. The maximization of the production of this compound was used as case study since it has several industrial applications, including synthetic chemistry using the enzymatic Aldol Syntheses[5] [18].

The model of the central carbon metabolism of *Escherichia coli* consists of mass balance equations for extra-cellular glucose and for intracellular metabolites. The mass balances take the following form:

$$\frac{dC_i}{dt} = \sum_j v_{ij} r_j - \mu C_i \tag{1}$$

Where C_i represents the concentration of metabolite i, μ is the specific growth rate and v_{ij} is the stoichiometric coefficient for this metabolite in reaction j, the rate of which is r_j.

3.2 Optimization Tasks and Algorithms

In this section, the optimization tasks and techniques employed are described. Two distinct scenarios were studied in this work, both using the model aforementioned. In the first, the problem at hand consists in determining the optimal knockout set that maximizes the production of a given metabolite (in this case DHAP) along a given time interval (in this case is was set to [0, 20] seconds). Therefore, the fitness function consists on the numerical integration of the target metabolite's concentration. The integration of the ODE is performed using the method provided by CVODE (that is suitable for both stiff and non-stiff ODE problems) with a step size of 0.1. In the simulations, the initial values for the model variables (i.e. initial concentrations) were set to the values supplied by [2].

In both cases, since the algorithms are stochastic, the optimization process was run for 30 times and the results are the means, presented within a 95% confidence interval.

In the first task, an EA with a set-based representation was used [16], where an individual encodes a subset of the full set of reactions in the model. To evaluate each solution, a simulation is run where the model is changed by removing all reactions encoded in the individual's genome. The fitness function is therefore calculated using this modified model.

It should be mentioned that the representation used in the EA employs a variable-sized genome, therefore allowing the competition of knockout sets with distinct cardinalities within the same population. Within this EA, the following reproduction operators are used to breed new individuals [16]:

- Grow mutation: consists in the introduction of a number of new elements into the set, whose values are randomly generated within the available range, avoiding duplicates;
- Shrink mutation: a number of randomly selected elements are removed from the set;
- Random mutation: replaces an element of the set by another, randomly generated in the allowed range; and,
- Modified Uniform crossover: it is inspired on the traditional uniform crossover operator and works as follows: the genes that are present in both parent sets are kept in both offspring; the genes that are present in only one of the parents are sent to one of the offspring, selected randomly with equal probabilities.

The following steps present the general structure of the EA:

1. Generate a population of NP individuals. Each individual represents a potential solution to the problem, initially created randomly.
2. For each individual in the population, evaluate its fitness by running the correspondent model simulation and computing the fitness function. If the stopping criteria is met, the algorithm stops and returns the best solution found.
3. Selection: First the set of E best individuals is copied to the next generation (elitism). Afterwards, a pool of $NP/2$ individuals (parents) is created using a *roulette wheel* scheme.
4. Reproduction: The set of available reproduction operators (crossover and mutation) are applied to the selected pool of parents, in order to generate the offspring ($NP/2$ new individuals are created that are inserted into the new generation). All reproduction operators available have the same probability of being chosen to breed each new individual.
5. The new population is completed by selecting $NP/2 - E$ individuals from the original population (a substitution rate of 50% is adopted). Return to step 2.

The EA was ran for 500 iterations with a population of 100 individuals. An elitism value of $E = 1$ is used.

In the second scenario, a similar approach was taken, but instead of finding a knockout set, the purpose is to modify the value of one of the kinetic

parameters of each reaction, in this case the v_{max}. The v_{max} parameter represents the maximum enzyme reaction rate under the conditions of the experiment. This value can be changed in a wet lab by changing the level of expression of certain enzymes in the re-engineered microbial strains.

In this second scenario, a DE algorithm was employed. The individuals encode the level of change for the v_{max} parameter of each reaction, when compared to the base value present in the original model. The level of change can vary between 0 and 2; a value of 1 means the parameter remains unchanged.

In this work, a variant of the DE algorithm called *DE/rand/1* was considered that uses a binomial crossover [17]. In this case, the following scheme is followed, in every generation, for each individual i in the population:

1. Randomly select 3 individuals r_1, r_2, r_3 distinct from i;
2. Generate a trial vector based on: $\boldsymbol{t} = \boldsymbol{r}_1 + F \cdot (\boldsymbol{r}_2 - \boldsymbol{r}_3)$;
3. Incorporate coordinates of this vector with probability CR;
4. Evaluate the candidate and use it in the new generation if it is at least as good as the current individual.

The DE was ran for 500 iterations with a population of 20 individuals. The F parameter was set to 0.5 and CR to 0.6.

4 Results

4.1 Gene Deletion Scenario

In Table 1 we show the results for the gene deletion task. The mean and confidence interval of the fitness function value (DHAP production) obtained for the best solution in each run is shown, as well as the mean number of knockouts. On the other hand, Figure 3 shows the histogram of the reaction knockouts, i.e. the number of times a given reaction is knocked-out in the best solution for the 30 runs.

Table 1. Results for the gene deletion task

Total Number Of Runs	30
Mean of fitness function (mM.s)	36.268 ± 2.8E-14
Mean Number Of Knockouts	15.2 ± 3.6
Confidence Level	95%

4.2 Kinetic Parameter Optimization

In Table 2 the results obtained for the v_{max} parameter optimization scenario are shown. Figure 4 shows the boxplot concerning the level of change in the v_{max} parameter for a number of reactions. To better compare with the previous experiment, this set is composed of the reactions that were most frequently the target of a knockout.

Table 2. Results for the kinetic parameter optimization task

Total Number Of Runs	30
Mean of fitness function (mM.s)	77.011 ± 3.88
Confidence Level	95%

4.3 Discussion

Regarding the first task, it is interesting to note that none of the main reactions that lead to the production of DHAP (PTS, PGI, PFK and ALDO) are knocked out in any of the best solutions for each run. The reactions that may impact negatively in the production of DHAP, even if not directly, have a higher chance of being knocked out as it can be seen in Figure 3. This result validates the proposed approach.

However, it is important to mention that the obtained knockout sets are not likely to produce viable mutants due to the fact that there are no restrictions regarding the set of possible reactions to inactivate. The reaction set composed by G6PDH, TIS, G3PDH and R5PI is always deleted in all the best individuals, thus inhibiting metabolic pathways like nucleotide and glycerol synthesis and thus suppressing biomass formation.

During the v_{max} parameter optimization, the best solutions lead to an increased production of DHAP. This is explained by the fact that, when tuning the v_{max} parameter for each reaction, we are allowing the reactions to have a reduced activity (if the reduction factor is 0 the reaction can even be knocked out as before) or an increased activity (the v_{max} value can be doubled). This provides much more flexibility and leads to higher values of the fitness function. Also, in contrast with the previous scenario, the mutants are more likely to be viable because most of the metabolic pathways are not completely inactivated.

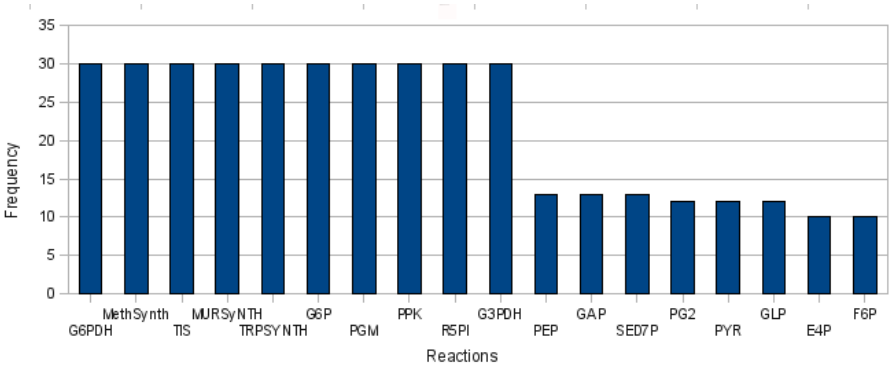


Fig. 3. Knockout frequency graph: only the reactions that have a frequency of at least 10 are shown

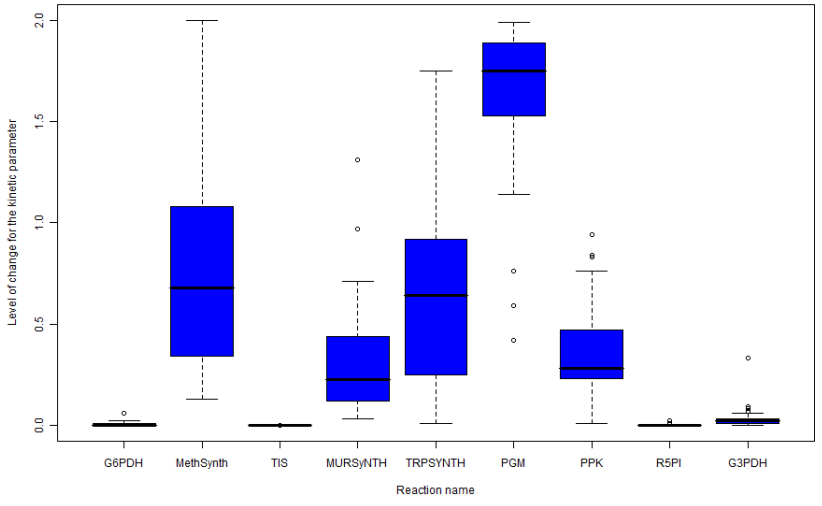


Fig. 4. Level of change for the v_{max} parameter in a selected set of reactions (selected from the set of reactions that were frequently knocked out in the previous experiment)

The v_{max} values obtained in the best solutions typically show an inactivation of the reactions unnecessary to the production of DHAP, resembling the knock-out sets produced in the first scenario, as it is obvious comparing the results of Figure 4 and Figure 3. The v_{max} parameter value for the PGM reaction is the only one that increases its base level value. This is due to the fact that it is the only parameter that lies in the denominator of the kinetic expression. All other parameters under optimization are being multiplied by the numerator of the reaction rate laws.

The optimization results for the studied scenarios emphasize the complex interactions involved, even in this very simple model. The methods presented merely serve as a proof of concept, since most of the solutions are likely to be biologically non-viable. Those limitations of the approach are mainly due to the nature of the models (the used model is known to be incomplete) and the constraints imposed over the solutions in these experiments.

If a more complete model is used and the constraints are biologically correct, the proposed framework can be used to reach biologically meaningful results. For example, is this case, to use this approach in a real metabolic engineering approach, some of the reactions would have to be constrained not to be a target for deletion and the limits over the v_{max} parameters would have to be carefully imposed.

5 Conclusions and Further Work

In recent years, several methods have been developed *in silico* with the purpose of identifying and characterizing microrganisms' metabolic functioning. So far, research has been mostly confined to explore parameter estimation problems,

based on fitting experimental data. On the other hand, Metabolic Engineering related approaches are based in steady state models. This study focus on studying novel ways of exploring dynamical models to optimize model modifications (e.g. model structure or parameter values) in different settings using as objective function the maximization of the production of a given metabolite of industrial interest.

The modular architecture of the proposed framework allows to replace any component of the dynamical model. For instance, when the rate law of a reaction has an unknown mathematical expression for a given model it can be replaced by a model built based on experimental data (e.g. a trained neural network).

In future work, the main issues to be tackled are the validation of this framework with other real-world case studies and also to make the computational tools available to the research community by integrating them in a proper platform with appropriate graphical user interfaces.

Regarding the optimization layer, a number of other algorithms have to be integrated in the framework, namely Genetic Programming [11] and Artificial Immune Systems [8] should be considered. The use of multi-objective optimization algorithms [15] in the optimization layer is also a promising route.

References

1. Bornstein, B.J., Keating, S.M., Jouraku, A., Hucka, M.: LibSBML: an API Library for SBML. Bioinformatics 24(6), 880–881 (2008)
2. Chassagnole, C., Noisommit-Rizzi, N., Schmid, J.W., Mauch, K., Reuss, M.: Dynamic modeling of the central carbon metabolism of *Escherichia coli*. Biotechnology and Bioengineering 79(1), 53–73 (2002)
3. Cohen, S., Hindmarsh, C.: Cvode, a stiff/nonstiff ode solver in c. Computers in Physics 10(2), 138–143 (1996)
4. Edwards, J.S., Palsson, B.O.: Metabolic flux balance analysis and the in silico analysis of escherichia coli k-12 gene deletions. BMC Bioinformatics 1(1), 1 (2000)
5. Gefflaut, T., Lemaire, M., Valentin, M.-L., Bolte, J.: A novel efficient synthesis of dihydroxyacetone phosphate and bromoacetol phosphate for use in enzymatic aldol syntheses. The Journal of Organic Chemistry 62(17), 5920–5922 (1997)
6. Haunschild, M.D., Freisleben, B., Takors, R., Wiechert, W.: Investigating the dynamic behavior of biochemical networks using model families. Bioinformatics 21(8), 1617–1625 (2005)
7. Hucka, M., Finney, A., Sauro, H.M., Bolouri, H., Doyle, J.C., Kitano, H., Arkin, A.P., Bornstein, B.J., Bray, D., Cornish-Bowden, A., Cuellar, A.A., Dronov, S., Gilles, E.D., Ginkel, M., Gor, V., Goryanin, I.I., Hedley, W.J., Hodgman, T.C., Hofmeyr, J.-H., Hunter, P.J., Juty, N.S., Kasberger, J.L., Kremling, A., Kummer, U., Le Novere, N., Loew, L.M., Lucio, D., Mendes, P., Minch, E., Mjolsness, E.D., Nakayama, Y., Nelson, M.R., Nielsen, P.F., Sakurada, T., Schaff, J.C., Shapiro, B.E., Shimizu, T.S., Spence, H.D., Stelling, J., Takahashi, K., Tomita, M., Wagner, J., Wang, J.: The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics 19(4), 524–531 (2003)
8. Ishida, Y.: The immune system as a self-identification process: A survey and a proposal. In: Proceedings of the IMBS 1996 (1996)

9. JFreeChart, http://www.jfreechart.org/jfreechart
10. Kitagawa, J., Iba, H.: Identifying metabolic pathways and gene regulation networks with evolutionary algorithms. Morgan Kaufmann, San Francisco (2003)
11. Koza, J.R.: Genetic programming (1992)
12. Mendes, P., Kell, D.: Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. Bioinformatics 14(10), 869–883 (1998)
13. Nummela, J., Julstrom, B.A.: Evolving petri nets to represent metabolic pathways. In: GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computation, pp. 2133–2139. ACM, New York (2005)
14. Patil, K., Rocha, I., Forster, J., Nielsen, J.: Evolutionary programming as a platform for in silico metabolic engineering. BMC Bioinformatics 6(308) (2005)
15. Maia, P., Ferreira, E.C., Rocha, I., Rocha, M.: Evaluating evolutionary multiobjective algorithms for the in silico optimization of mutant strains. In: 8th IEEE International Conference on BioInformatics and BioEngineering Workshops (BIBE 2008), Athens, Greece (2008)
16. Rocha, M., Maia, P., Mendes, R., Ferreira, E.C., Patil, K., Nielsen, J., Rocha, I.: Natural computation meta-heuristics for the in silico optimization of microbial strains. BMC Bioinformatics 9(499) (2008)
17. Storn, R., Price, K.: Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11, 341–359 (1997)
18. Thiem, J.: Applications of enzymes in synthetic carbohydrate chemistry. FEMS Microbiology Reviews 16(2-3), 193–211 (1995)
19. Wang, Q.J.: Using genetic algorithms to optimise model parameters. Environmental Modelling and Software with Environment Data News 12(8), 27–34 (1997)
20. Yang, K., Ma, W., Liang, H., Ouyang, Q., Tang, C., Lai, L.: Dynamic simulations on the arachidonic acid metabolic network. PLoS Computational Biology, e55.eor+ (February 2007) (preprint)

Clustering Metagenome Short Reads Using Weighted Proteins

Gianluigi Folino[1], Fabio Gori[2], Mike S.M. Jetten[2], and Elena Marchiori[2,⋆]

[1] ICAR-CNR, Rende, Italy
[2] Radboud University, Nijmegen, The Netherlands
elenam@cs.ru.nl

Abstract. This paper proposes a new knowledge-based method for clustering metagenome short reads. The method incorporates biological knowledge in the clustering process, by means of a list of proteins associated to each read. These proteins are chosen from a reference proteome database according to their similarity with the given read, as evaluated by BLAST. We introduce a scoring function for weighting the resulting proteins and use them for clustering reads. The resulting clustering algorithm performs automatic selection of the number of clusters, and generates possibly overlapping clusters of reads. Experiments on real-life benchmark datasets show the effectiveness of the method for reducing the size of a metagenome dataset while maintaining a high accuracy of organism content.

1 Introduction

The rapidly emerging field of metagenomics seeks to examine the genomic content of communities of organisms to understand their roles and interactions in an ecosystem. Given the wide-ranging roles microbes play in many ecosystems, metagenomics studies of microbial communities will reveal insights into protein families and their evolution. Because most microbes can not grow in the laboratory using current cultivation techniques, scientists have turned to cultivation-independent techniques to study microbial diversity.

At first shotgun Sanger sequencing was used to survey the metagenomic content, but nowadays massive parallel sequencing technology like 454 or Illumina, allow random sampling of DNA sequences to examine the genomic material present in a microbial community [5]. Using metagenomics, it is now possible to sequence and assemble genomes that are constructed from a mixture of organisms.

While it is common to refer to the genome sequence as if it were a single, complete and contiguous DNA string, it is in fact an assembly of billions of small, partially overlapping DNA fragments.

For a given sample, one would like to determine the phylogenetic provenance of the obtained fragments, the relative abundance of its different members, their metabolic capabilities, and the functional properties of the community as a

⋆ Contact author.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 152–163, 2009.

whole. To this end, computational analyses are becoming increasingly indispensable tools [11,14].

Sophisticated computer algorithms (assemblers and scaffolders) merge these DNA fragments into contigs, and place these contigs into sequence scaffolds using various methods and tools (cf., e.g., [2,6,4]). Clustering methods are used for rapid analysis of sequence diversity and internal structure of the sample [8], for discovering protein families present in the sample [3], and as a pre-processing set for performing comparative genome assembly [13], where a reference closely related organism is employed to guide the assembly process.

In this paper we focus on the problem of clustering metagenome short reads [3,8]. Our approach is inspired by a recent work by Dalevi et al [3], where a method for clustering reads is proposed based on a set of proteins, called *proxygenes*, obtained by BLASTx of the reads against the protein sequences of a reference database. However, the results of the method proposed in [3] depend on the read selected at the beginning of the procedure, and reads clusters are not allowed to overlap.

We propose a new robust method for clustering metagenome short reads based on weighted proteins. The method generates a set of clusters, where each cluster is represented by one *proxygene*. This method has the following desirable features:

- It incorporates biological knowledge in the clustering process;
- It performs automatic selection of the number of clusters;
- It generates potentially overlapping clusters of reads.

Specifically, the proposed method consists of three main steps.

First, it uses a specialized version of BLAST (Basic Local Alignment Search Tool), called BLASTx, for associating a list of hits to each read. Each hit consists of one protein, two score values, called bit and identities, which measure the quality of the read-protein matching, and one confidence value, called E-value, which amounts to a confidence measure of the matching.

Next, a maximum of K proteins for each read, among those having E-value smaller than a given threshold α are selected. The selected proteins are weighted by means of a novel measure based on the bit- and identity- scores, which assigns small weights to proteins of high average quality.

Finally, the reads are clustered by translating the clustering problem into an instance of the weighted set covering problem (`WSC`). The `WSC` is a popular constrained optimization problem used in many real-life applications. Given a set of weighted columns and a set of rows, where each row is covered by at least one column, the `WSC` problem amounts to find a set of columns covering all the rows and having minimum total weight. In our context, columns are proteins and rows are reads. A protein covers a read if it belongs to the set of the selected hits of that read. We employ a publicly available fast heuristic algorithm for the weighted set covering problem [10]. The resulting clustering method generates a set of clusters, where each cluster is represented by one protein called *proxygene*.

While in [3] the proxygenes of a cluster is selected within a set of proteins associated to it, in our method clustering and proxygene selection are performed at the same time.

In order to assess the effectiveness and benefits of the proposed clustering method, we consider the metagenomic datasets recently introduced in [3]. We measure the quality of the resulting clusters by means of the organism content of the clusters [8], their size, number and overlapping.

Specifically, we analyse the behavior of the clustering algorithm when varying its parameters K and α. Results show that the number of clusters decreases when bigger values of K are chosen while their overlapping increases. The organism content of the clusters does not change substantially for higher values of K and small α (0.01), indicating the effectiveness of the proposed approach in reducing the size of a metagenome dataset while maintaining a high accuracy of organism content.

The proposed method can therefore be used for reducing the size of the dataset while maintaining accuracy of functional and taxonomic content of a metagenome, and for discovering knowledge related to the protein content and the taxonomic organization of the organisms contained in the sample.

In general, the results substantiate the effectiveness of the proposed clustering method for mining metagenomic datasets.

2 Clustering Metagenome Short Reads

Clustering analysis for metagenomics amounts to group similar partial sequences, such as raw sequence reads, or candidate ORF (Open Reading Frame) sequences generated by an assembly program into clusters in order to discover information about the internal structure of the considered dataset, or the relative abundance of protein families. Different methods for clustering analysis of metagenomic datasets have been proposed, which can be divided into two main approaches: sequence- and evidence-based methods. *Sequence-based* methods compare directly sequences using a similarity measure either based on sequence overlapping [8] or on extracted features such as oligonucleotide frequency [2]. *Evidence-based* methods employ knowledge extracted from external sources in the clustering process, like proteins identified by a BLASTx search (proxygenes) [3].

Here we use the latter approach for clustering short reads. Specifically, we propose a clustering method consisting of the following main steps:

1. Run BLASTx on the reads;
2. Assign weights to proteins resulting from BLASTx;
3. Cluster the reads using the weighted proteins obtained from the previous step as candidate cluster prototypes.

The result is a set of possibly overlapping clusters of reads, where each cluster is represented by a protein. The number of clusters is automatically determined by the algorithm. The proposed method has just two parameters: the maximum number K of hits selected for each read, and the E-value threshold α. Below the steps of the method are described in detail.

3 Run BLASTx on the Reads

The knowledge used by the proposed clustering algorithm is extracted by a reference proteome database by matching reads to that database by means of BLASTx, a powerful search program. BLASTx belongs to the BLAST (Basic Local Alignment Search Tool) family, a set of similarity search programs designed to explore all of the available sequence databases regardless of whether the query is protein or DNA [7,9]. BLASTx is the BLAST program designed to evaluate the similarities between DNA sequences and proteins; it compares nucleotide sequence queries dynamically translated in all six reading frames to peptide sequence databases. The scores assigned in a BLAST search have a statistical interpretation, making real matches easier to distinguish from random background hits. In the following we summarize the main features of BLAST.

BLAST uses a heuristic algorithm that seeks local as opposed to global alignments and is therefore able to detect relationships among sequences that share only isolated regions of similarity [1]. When a query is submitted, BLAST works by first making a look-up table of all the *words* (short subsequences, three letters in our case) and *neighboring words*, i.e., similar words in the query sequence. The sequence database is then scanned for these strings; the locations in the databases of all these words are called *word hits*. Only those regions with word hits will be used as alignment seeds. When one of these matches is identified, it is used to initiate gap-free and gapped extensions of the word. After the algorithm has looked up all possible words from the query sequence and extended them maximally, it assembles the statistically significant alignment for each query-sequence pair, called *High-scoring Segment Pair* (HSP).

The matching reliability is evaluated trough *Bit Score*, denoted by S_B, and *E-value*, denoted by E. The *bit score* of one HSP is computed as the sum of the scoring matrix values for that segment pair. The *E-value* is the number of times one might *expect* to see such a query-sequence match (or a better one) merely by chance.

Another score very important for BLASTx is *Identities score*, defined as the proportion of the amino-acids in the database sequence that are matched by the amino-acids translation of the current query frame.

We refer to [7] for a formal description of these measures.

In our method, the E-value is used to constrain the number of output hits, while the bit and identities scores are used to weight proteins as follows.

4 Assign Weights to Proteins

From each hit that BLASTx outputs for a given read r, we extract a 4-dimensional vector $h = (p, S_B, Id, E)$ where p is the matched protein, S_B the bit score, Id the identities score, and E the E-value of that match. With abuse of notation we refer to such a vector as 'the *hit* of r'.

For a read r let $Hit_{\alpha,r}$ be the sequence, sorted in increasing order of E-values, of its hits having E-value smaller than a given threshold α. Denote by $r_1, \ldots, r_m$

the set of reads r with non-empty $Hit_{\alpha,r}$. Let K_{α,r_i} be the sequence of the first K elements of Hit_{α,r_i} (the entire sequence if K exceeds the sequence's length). We write K_i instead of K_{α,r_i} when no ambiguity arises.

Let $P = \{p_1, \ldots, p_n\}$ be the set of proteins occurring in $\cup_{i=1}^{m} K_i$. For each protein $p \in P$ define the set

$$H_p := \{h \in \cup_{i=1}^{m} K_i \mid h(1) = p\},$$

where $h(1)$ denotes the first component of the hit vector h. Thus H_p consists of the selected hits containing p.

Define the weight of p as follows:

$$w_p := 1 + \ulcorner \frac{1}{|H_p|} \sum_{h \in H_p} (100 \frac{\texttt{max\_score} - S_B(h)}{\texttt{max\_score} - \texttt{min\_score}} + 100 - Id(h)) \urcorner,$$

where $\ulcorner v \urcorner$ denotes the smallest integer bigger or equal than v, $S_B(h)$ and $Id(h)$ are the bit- and identity-score of h, respectively, and $|H_p|$ the cardinality of H_p. By construction weights are positive integers between 1 and 201.

The bit score has been used e.g. in [3] to define a measure of protein relevance.

Our approach for scoring proteins differs from e.g. the one used in [3] in two main ways. First, we score proteins using also the identities score. Second, we score each protein globally, by considering all the hits involving that protein, while in [3] proteins are score locally after the reads have been clustered. Specifically, in the latter approach, first reads and proteins are clustered together, and only at the end of the clustering process, each protein of a cluster is scored by means of the cumulative bit score of its alignment to the reads within the same cluster.

5 Clustering Reads Using Weighted Proteins

The clustering algorithm selects a set of cluster representatives from P, whose union covers all the considered reads, and with minimum total weight. The clusters are generated by a fast heuristic algorithm for the weighted set covering problem [10] applied to the m selected reads and the set P of proteins weighted as described above. The number of clusters is automatically computed by the procedure.

Formally, consider the vector of protein weights $w \in \mathbb{N}^n$ and the matrix $A \in \{0,1\}^{m \times n}$ whose elements a_{ij} are such that:

$$a_{ij} = \begin{cases} 1, & \text{if column } j \text{ covers row } i, \\ 0, & \text{otherwise.} \end{cases}$$

So a row i is covered by a column j if a_{ij} is equal to 1. In the context of our application $a_{ij} = 1$ if protein p_j occurs in the set K_i of selected hits of read r_i.

The *weighted set covering problem* (WSC, in short) can be formulated as a constrained optimization problem as follows:

$$\min_{x\in\{0,1\}^n} \sum_{j=1}^{n} x_j w_j, \qquad \text{such that } \sum_{j=1}^{n} a_{ij} x_j \geq 1, \qquad \text{for} \quad i = 1, \ldots, m. \quad \text{(WSC)}$$

The variable x_j indicates whether column j belongs to the solution ($x_j = 1$) or not ($x_j = 0$). The m constraint inequalities are used to express the requirement that each row i be covered by at least one column (that is, for each read r_i, at least one protein in K_i is chosen). The weight w_j specifies the cost of column j.

The Weighted Set Covering problem is one of the oldest and best studied NP-hard problems. It has been successfully employed to tackle real-life problems in diverse domains, including biology (cf., e.g., [15]). Here we use a fast heuristic algorithm for WSC[1] originally developed for tackling airline crew scheduling problems [10].

A solution corresponds to a subset of P consisting of those proteins p_j such that $x_j = 1$. Each of the selected proteins is a *proxygene*. It represents a cluster containing those reads r having that protein in K_r.

Example 1. We illustrate this process by means of a toy example (cf. Figure 1). Suppose given a set of five reads $\{r_1, \ldots, r_5\}$ and suppose that the proteins occurring in the selected hits of these reads are:

- $\{p_1, p_3, p_5\}$ for read r_1;
- $\{p_1, p_3, p_6\}$ for read r_2;
- $\{p_2, p_5\}$ for read r_3;
- $\{p_2\}$ for read r_4;
- $\{p_2, p_3, p_6\}$ for read r_5.

Assume for the sake of simplicity that all proteins have equal weight. Then Figure 1 (left part) shows the corresponding 5-row, 6-column matrix a_{ij}. The WSC-clustering algorithm applied to this problem instance outputs the set $\{p_2, p_3\}$ of columns, having total weight equal to 2 (see Figure 1 right part). The selected columns correspond to the two clusters $\{r_3, r_4, r_5\}$ and $\{r_1, r_2, r_5\}$, respectively.

6 Experiments

We consider three complex metagenome datasets introduced in [3], called in the following M1, M2 and M3. These datasets were generated, respectively, from 9, 5 and 8 genome projects, sequenced at the Joint Genome Institute (JGI) using the 454 GS20 pyrosequencing platform that produces $\sim$ 100 bp reads. From each genome project, reads were sampled randomly at coverage level 0.1X. The coverage is defined as the average number of times a nucleotide is sampled. This resulted in a total of 35230, 28870 and 35861 reads, respectively.

Table 1 shows the names of the organisms and the number of reads generated for the M1 dataset. The reader is referred to [3] for a detailed description of all the datasets.

[1] Publicly available at http://www.cs.ru.nl/~elenam

	p1	p2	p3	p4	p5	p6
r1	1	0	1	0	1	0
r2	1	0	1	0	0	1
r3	0	1	0	0	1	0
r4	0	1	0	0	0	0
r5	0	1	1	0	0	1

	p1	p2	p3	p4	p5	p6
r1	1	0	1	0	1	0
r2	1	0	1	0	0	1
r3	0	1	0	0	1	0
r4	0	1	0	0	0	0
r5	0	1	1	0	0	1

Fig. 1. Left: input covering matrix; position (i,j) contains a 1 if protein p_j occurs in the set of selected hits of read r_i, otherwise it contains a 0. Right: the proteins selected by the WSC-clustering algorithm are indicated by arrows.

Table 1. Characteristics of the organisms used in the experiments: the identifier and name of the organism, the size of its genome and the total number of reads sampled (M1 dataset).

Id.	Organism	genome size (bp)	reads sampled
a	Clostridium phytofermentans ISDg	4 533 512	4638
b	Prochlorococcus marinus NATL2A	1 842 899	1866
c	Lactobacillus reuteri 100-23	2 174 299	2371
d	Caldicellulosiruptor saccharolyticus DSM 8903	2 970 275	2950
e	Clostridium sp. OhILAs	2 997 608	2934
f	Herpetosiphon aurantiacus ATCC 23779	6 605 151	6937
g	Bacillus weihenstephanensis KBAB4	5 602 503	4158
h	Halothermothrix orenii H 168	2 578 146	2698
i	Clostridium cellulolyticum H10	3 958 683	3978

In our experiments we use the NR[2] (non-redundant) protein sequence database as reference database for BLASTx. The parameters of the external software we used are set as follows. For BLASTx the default parameters were used. In all experiments WSCP was run with pre-processing ($-p$), number of iterations equal to 1000 ($-x1000$), one tenth of the best actual cover used as starting partial solution ($-a0.1$), and 150 columns to be selected for building the initial partial cover at the first iteration ($-b150$). For lack of space, we refer to [10] for a detailed description of the WSCP program.

6.1 Evaluation

First of all we set α to a reasonable value, equal to 0.01, and disregard all the reads with E-value greater than α, resulting in the selection of 21236 reads for M1, 21064 for M2 and 24043 for M3, respectively. We analyse the clusterings obtained by varying the value of K by means of the following characteristics.

– The number of clusters obtained, their size and overlapping;

[2] Publicly available at ftp://ftp.ncbi.nlm.nih.gov/blast/db.

Table 2. Summary of the results of experiments for α=0.01 and varying K (M1 dataset)

K	1	2	10	50	1000
number of proteins selected	13594	19967	66005	174110	360578
number of clusters	13594	13197	12599	12091	11763
number of singleton clusters	9003	8334	7420	6666	6145
maximum size of clusters	17	21	23	28	32
total size of overlapping	0	273	877	1640	2979

- The reduction factor, defined as the number of selected reads divided by the number of clusters;
- The homogeneity of the clusters as measured by the so-called *cluster purity*, defined as the maximum fraction of its elements belonging to the same organism, that is

$$purity(C) := \frac{1}{|C|} \max_{i=1,\ldots,n_{org}} (|C|_{organism=i}),$$

where $|C|_{organism=i}$ denotes the number of elements of cluster C belonging to organism i, and n_{org} the number of organisms.

A similar analysis is performed by fixing K to a reasonable value, equal to 50, and varying the threshold α.

Figure 2 shows in more detail the trends of the cluster homogeneity and of the reduction factor.

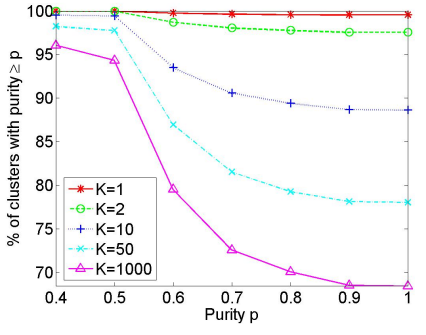
6.2 Results: Fixed Value of α=0.01 and Varying K Values

Table 2 summarizes the results with this parameter setting for dataset M1. The number of selected proteins increases when K increases, while the number of clusters decreases, indicating effectiveness of the method to select few proxy-genes. Furthermore, the number of singleton clusters also decreases for higher values of K, indicating a stronger bias towards the grouping of reads. A similar trend can be observed for datasets M2 and M3 (results omitted for lack of space).

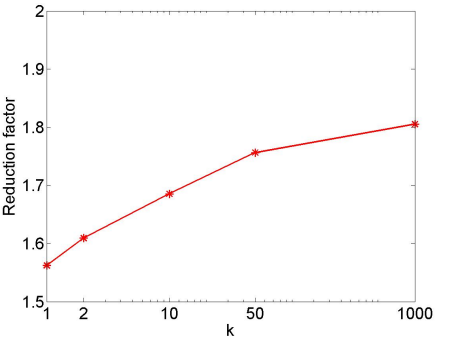
Figures 2 (a), (c) and (f) show, for each datasets, the percentage of non-singleton clusters having purity greater or equal than a given value p, for selected values of p in $[0.4, 1]$. For all the datasets, the curve at $K = 1$ dominates all other ones, justified by the fact that the corresponding clustering contains many clusters of small size, which are likely to have higher purity. For instance, for the M1 dataset with $K = 1$ and 1000, about 75% and 35% of the clusters have size equal to 2, respectively.

Figures 2 (b,d and f) show, for each dataset, the reduction factor for different values of K. As expected, a larger value of K results into a higher reduction factor.

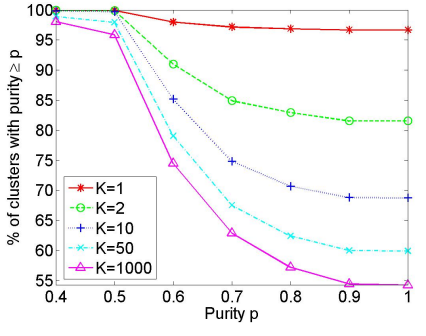
Finally, Figure 3 shows, for dataset M1, how the number of reads occurring in more than k clusters varies for different choices of K. For a small value of K



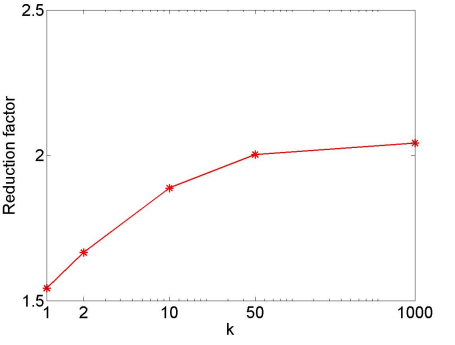
(a)



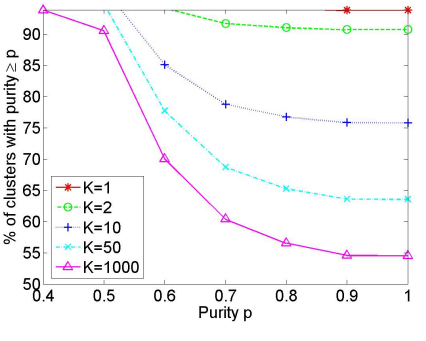
(b)



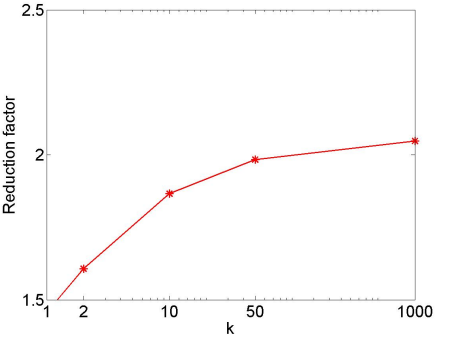
(c)



(d)



(e)



(f)

Fig. 2. (a),(c),(e): plots of percentage of non-singleton clusters with purity $\geq p$ for different values of K, for M1, M2 and M3, respectively. (b),(d),(f): plots of the reduction factor for different values of K, for M1, M2 and M3, respectively. Right: plot of the number nr of non-singleton clusters of size s for $K = 1, 1000$.

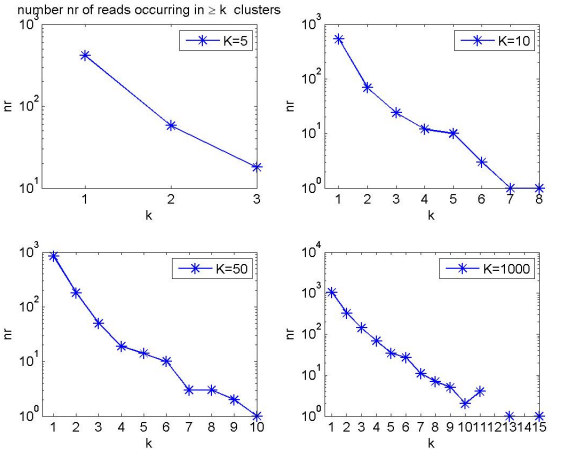


Fig. 3. Plot of the number nr of reads occurring in k clusters (M1 dataset)

(equal to 5) a read occurs in at most 3 clusters, while for a very high value of K (equal to 1000) a read occurs in at most 15 clusters. Indeed, as reported in Table 2, the total overlapping shows a substantial increase for high values of K, where by total overlapping we mean the sum of cardinality of the clusters minus the number (21236) of selected reads. These results can be justified by the fact that K is an upper bound on the maximum number of clusters one read may belong to. Similar results, not showed for lack of space, were obtained using M2 and M3 datasets.

6.3 Results: Fixed Value of $K = 50$ and Varying α Values.

Table 3 summarizes the results with this parameter setting for dataset M1. Higher values of α result into the selection of an higher number of reads and of proteins. Moreover, clusters of bigger size and overlapping are obtained.

The plots of Figure 4 show that on the M1 dataset, small α values lead to clusterings where 90% of the clusters are very accurate, in terms of organism

Table 3. Summary of the results of experiments for $K = 50$ and varying α (M1 dataset)

α	0.1	0.01	0.001	0.0001	1e-006
number of reads selected	22219	21236	20300	19085	16736
number of proteins selected	208443	174110	146524	116682	72149
number of clusters	12283	12091	11850	11534	10660
number of singleton clusters	6464	6666	6772	6889	6801
maximum size of clusters	30	28	27	23	19
total size of overlapping	2026	1640	1326	1066	528

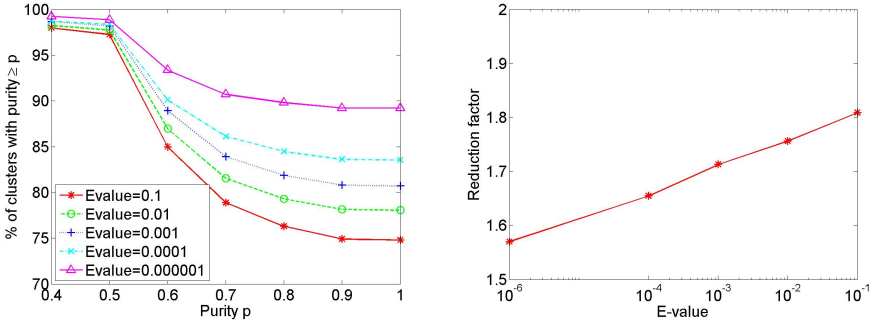


Fig. 4. a) plot of percentage of non-singleton clusters with purity $\geq p$ for different values of α. b) plot of the reduction factor for different values of α (M1 dataset)

content, and a reduction factor of about 1.6. For higher values of α clusters purity decreases reaching a minimum of about 75%, while reduction factor increases reaching a maximum of about 1.8. Similar results are obtained for datasets M2 and M3. Thus, the user can decide a tradeoff between purity and reduction, depending on the specific research question to be addressed.

7 Conclusion and Future Work

This paper introduced a new evidence-based method for clustering metagenome short reads and analysed its performance on benchmark metagenome datasets. Results indicated effectiveness of the proposed method as a tool for mining metagenome data.

We focussed on the experimental analysis of the two parameters of the proposed clustering method, K and α. As for the computational cost, the WSC-clustering algorithm is very efficient, due to the fast heuristic employed to search for an optimal set cover. However, the extraction of the hits from the initial dataset of reads is computationally expensive. Nevertheless, the latter process can be parallelized by partitioning the reads and running BLASTx independently on each group of the partition.

In the future, we intent to investigate in more depth the biological meaning of the resulting clusters, in particular their functional and taxonomic content, in order to discover knowledge related to the protein content and the taxonomic organization of the organisms contained in metagenomes.

Furthermore, it is interesting to investigate if the clusterings obtained by varying the value of such parameters could be used for analyzing the dynamics of organism grouping, as modeled by the protein-based clustering, in particular whether such model of organism-grouping dynamics is related to the taxonomic evolution of the corresponding metagenome sample.

Moreover, similar experiments on data with higher coverage will be performed, and analysis with more sophisticated measures of cluster homogeneity, like Normalized Mutual Information and Entropy Correlation Coefficient [12].

Acknowledgements. We would like to thank Mavrommatis Konstantinos for providing the datasets in [3] as well as useful information about such data.

References

1. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic local alignment search tool. J. Molecular Biology 215(3), 403–410 (1990)
2. Chan, C.K., Hsu, A.L., Tang, S., Halgamuge, S.K.: Using growing self-organising maps to improve the binning process in environmental whole-genome shotgun sequencing. Journal of Biomedicine and Biotechnology (2008)
3. Dalevi, D., Ivanova, N.N., Mavromatis, K., Hooper, S.D., Szeto, E., Hugenholtz, P., Kyrpides, N.C., Markowitz, V.M.: Annotation of metagenome short reads using proxygenes. Bioinformatics 24(16) (2008)
4. Mavromatis, K., et al.: Use of simulated data sets to evaluate the fidelity of metagenomic processing methods. Nature Methods 4(6), 495–500 (2007)
5. Yooseph, S., et al.: The sorcerer ii global ocean sampling expedition: Expanding the universe of protein families. PLoS Biol. 5(3), 432–466 (2007)
6. Hernandez, D., Francois, P., Farinelli, L., Osteras, M., Schrenzel, J.: De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. Genome Research 18(5), 802–809 (2008)
7. Korf, I., Yandell, M., Bedell, J.: BLAST. O'Reilly & Associates, Inc., Sebastopol (2003)
8. Li, W., Wooley, J.C., Godzik, A.: Probing metagenomics by rapid cluster analysis of very large datasets. PLoS ONE 3(10) (2008)
9. Madden, T.: The BLAST Sequence Analysis Tool, ch. 16. Bethesda, MD (2002)
10. Marchiori, E., Steenbeek, A.: An evolutionary algorithm for large scale set covering problems with application to airline crew scheduling. In: Real World Applications of Evolutionary Computing. LNCS, vol. 1083, pp. 367–381 (2000)
11. McHardy, A.C., Rigoutsos, I.: What's in the mix: phylogenetic classification of metagenome sequence samples. Current Opinion in Microbiology 10, 499–503 (2007)
12. Pluim, J.P.W., Antoine Maintz, J.B., Viergever, M.A.: Image registration by maximization of combined mutual information and gradient information. IEEE Trans. Med. Imaging 19(8), 809–814 (2000)
13. Pop, M., Phillippy, A., Delcher, A.L., Salzberg, S.L.: Comparative genome assembly. Briefings in Bioinformatics 5(3), 237–248 (2004)
14. Raes, J., Foerstner, K.U., Bork, P.: Get the most out of your metagenome: computational analysis of environmental sequence data. Current Opinion in Microbiology 10, 490–498 (2007)
15. Zhao, W., Fanning, M.L., Lane, T.: Efficient RNAi-based gene family knockdown via set cover optimization. Artificial Intelligence in Medicine 35(1-2), 61–73 (2005)

A Memetic Algorithm for Phylogenetic Reconstruction with Maximum Parsimony

Jean-Michel Richer[1], Adrien Goëffon[2], and Jin-Kao Hao[1]

[1] University of Angers, LERIA, 2 Bd Lavoisier, 49045 Anger Cedex 01, France
[2] LaBRI (UMR 5800) Université de Bordeaux 351 cours de la Libération 33405 Talence Cedex, France

Abstract. The Maximum Parsimony problem aims at reconstructing a phylogenetic tree from DNA, RNA or protein sequences while minimizing the number of evolutionary changes. Much work has been devoted by the research community to solve this NP-complete problem and many algorithms and techniques have been devised in order to find high quality solutions with reasonable computational resources. In this paper we present a memetic algorithm (implemented in the software Hydra) which is based on an integration of an effective local search operator with a specific topological tree crossover operator. We report computational results of Hydra on a set of 12 benchmark instances from the literature and demonstrate its effectiveness with respect to one of the most powerful software (TNT). We also study the behavior of the algorithm with respect to some fundamental ingredients.

Keywords: Maximum Parsimony, phylogeny, progressive descent, tree crossover, memetic algorithm.

1 Introduction

Phylogenetics, also known as Phylogenetic Systematics, is the formal name for the field within Biology that reconstructs evolutionary history of species. Phylogenetics studies the connections between groups of species (as understood by an ancestor / descendant relationships) which are represented by a phylogenetic tree. In such a phylogenetic tree, the leaves represent contemporary or extinct species and internal nodes correspond to hypothetical ancestors.

Recent advances in genomics, including whole-genome sequencing have mainly influenced phylogenetics. In the past, morphological characters (like size, color, number of legs, etc...) were used for inferring phylogenies. With today's data obtained from the sequencing programs, we can compare organisms from the information extracted from their genetic material (DNA, RNA) or their proteome (protein sequences). This is particularly useful when dealing with prokaryote organisms like bacteria or viruses for which morphological characters are difficult to define or identify. *Molecular* phylogenetics has then become an important field in Bioinformatics and finds many applications in biology and medicine like genetic evolution, taxonomy and classification, or virus detection and mutation [13].

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 164–175, 2009.

Much work has been devoted to the problem of phylogeny reconstruction since the 1960s. The general principle behind phylogenetic methods is to find a tree that minimizes sequence changes or mutations. We can bring three different approaches to light:

- *Distance methods*, inspired by the work of Sokal and Sneath on clustering [19], introduced in 1967 by [2] or by [6] rely on a matrix of distances observed between species. The most popular algorithm of this class of methods is probably the *Neighbor-Joining* from [17], improved by [8]. Those methods are very efficient for they rely on an algorithm of polynomial complexity but they sometimes lack robustness.
- *Probabilistic methods* are based on a model of evolution of characters. The Maximum Likelihood (ML), introduced by [4] in 1981 provides a general framework that consists in inferring the most probable phylogeny that maximizes the likelihood of observed sequences. Although ML is popular for phylogenetic inference because it is considered as a robust method, it is more computationally expensive than other methods.
- *Cladistic methods* are based on a matrix of given characters. The most well-known method of this class relies on the Maximum Parsimony (MP) criterion [3] as it is quite simple to apprehend. Such a method aims at building a binary tree that minimizes the number of changes without resorting to a particular model of evolution. The cost of a tree can be computed in polynomial time [5]. However, the search for an optimal tree is computationally intractable. Indeed, the Maximum Parsimony problem is extremely difficult to solve since it is equivalent to the NP-complete Steiner problem in the hypercube [7]. This is why heuristics methods constitute the main alternative in order to obtain a near optimal tree with reasonable computation time [12,15].

In this article, we are interested in finding high quality solutions for the phylogeny reconstruction problem with Maximum Parsimony under Fitch's criterion. We present a memetic algorithm which combines a local searcher using a parametric neighborhood relation and a specific tree crossover based on a topological distance. The crossover operator aims at creating new phylogenetic trees by conserving topological distance properties of parent trees while the local searcher improves the quality of each newly created offspring by a multi-neighborhood descent algorithm.

In order to accelerate the fitness evaluation of potential solutions (phylogenetic trees) which is a critical issue in our context, we introduce a vectorization technique to fully take advantage of some characteristics offered by modern x86 processors. To our knowledge, this technique was never described in the literature and helps decrease the computation time by about 80% on any x86 modern processor.

To assess the performance of the proposed memetic algorithm, we report computational results on a set of 12 benchmark instances from TreeBase, a well-known database on phylogenetic information and compare these results with those obtained by TNT, which is considered to be the most efficient software for the MP problem.

The rest of the paper is organized as follows. In the next section we formally introduce the MP problem. We then describe the main components of our memetic algorithm (Section 3). Computational results and comparisons are shown in Section 4 where an analysis is also given to study the influence of some basic ingredients on the performance of the algorithm (Section 4.4).

2 Phylogeny Reconstruction and Maximum Parsimony

Phylogeny generally takes as input a multiple alignment which is a matrix of characters composed of n lines (related to a set S of species, where $|S| = n$) and k columns which represent the characters of the sequences. Each sequence is also called a taxon (or taxonomic unit, plural taxa). Each character of the matrix belongs to an alphabet Σ. The aim of the Maximum Parsimony problem is to find a phylogenetic tree (i.e. generally a binary tree, rooted or unrooted) that minimizes the number of changes (or mutations) between sequences. Each leaf of the tree is associated to one of the n species and the cost (or the number of mutations) of the overall tree can be estimated by building hypothetical *sequences of parsimony* from the leaves toward the root of the tree. More precisely we can formulate the following definitions:

Definition 1 (Sequence of parsimony - see [5]). *Given two sequences S_1 and S_2 of length k characters such that $S_1 = x_1 \cdots x_k$, $S_2 = y_1 \cdots y_k$ with $\forall i \in \{1..k\}, x_i, y_i$ belong to the power set $\mathcal{P}(\Sigma)$, the sequence of parsimony of S_1 and S_2, noted $F(S_1, S_2) = z_1 \cdots z_k$ is obtained by :*

$$\forall i, 1 \leq i \leq k, z_i = \begin{cases} x_i \cup y_i, \text{ if } x_i \cap y_i = \emptyset \\ x_i \cap y_i, \text{ otherwise} \end{cases}$$

The cost (or number of mutations) of a sequence of parsimony is defined by:

$$\phi(F(S_1, S_2)) = \sum_{i=1}^{k} c_i \quad \text{where} \quad c_i = \begin{cases} 1, \text{if } x_i \cap y_i = \emptyset \\ 0, \text{otherwise} \end{cases}$$

Definition 2 (Rooted Binary Tree of Parsimony). *Let S be a set of n aligned sequences of length k where each character of the sequence is expressed over a given alphabet Σ. Let $T = (V, E)$ be a binary tree, where $V = \{v_1, \ldots, v_r\}$ is the set of nodes and $E \subseteq \{(u, v)/u, v \in V\}$ is the set of edges. T is called a binary tree of parsimony of S if :*

- *there exist $r = 2 \times n - 1$ nodes partitioned in two subsets:*
  - *I : a set of internal nodes composed of $n - 1$ nodes each having 2 descendants,*
  - *L : a set of leaves composed of n nodes with no descendant.*
- *there exists a bijection from the set of sequences S to the set of leaves L,*
- *each internal node w of I is assigned to a (hypothetical) sequence of parsimony $S_w = F(S_u, S_v)$.*

Definition 3 (Cost of a Tree of Parsimony). *Let T be a binary tree of parsimony of a set of sequences S. The cost (or score) of T, $\phi(T)$ is equal to* $\sum \phi(S_w), \forall w \in I$.

Remark: For a given phylogenetic tree, its parsimony score can be efficiently (in polynomial time) calculated by the Fitch's algorithm given in [5]. However, it is primordial to accelerate the scoring process as much as possible within a search algorithm when a great number of potential trees must be evaluated.

Definition 4 (Maximum Parsimony Problem - MP). *Given a set S of n sequences of length k, expressed over an alphabet Σ, find the most parsimonious tree $T^\star$ of S such that the score of parsimony of $T^\star$ is minimum.*

For a set of S of sequences, there are $\prod_{i=3}^{|S|}(2i-3)$ possible parsimony trees. The MP problem is thus a highly combinatorial search problem.

3 A Memetic Algorithm for the MP

3.1 Outline of the Hydra Algorithm

The Hydra algorithm presented in this paper follows a simple, yet powerful memetic schema. From an initial population of solutions, the algorithm carries out a number of evolution cycles. At each cycle, two parents are selected and a crossover operator is applied to the parents to create a new solution (Section 3.2). Each newly created offspring undergoes some local improvements (Section 3.4) before being inserted into the population. Other issues to be considered concern among others fitness evaluation (Section 3.3), parents selection and insertion condition. Parents selection operates with a tournament selection strategy. Two groups of 20% of the individuals are first constituted and two individuals that represent the best individuals of each group are then selected. A new individual T is inserted into the population if it is not already present in the population. The individual that will be removed from the population can be the oldest or the closest to the new one. The outline of the Hydra algorithm is given on Algorithm 1. We describe below the basic ingredients of this memetic algorithm for the MP.

3.2 Initial Population

The population is composed of phylogenetic trees (individuals) for the given set of taxa. The individuals of the initial population are generated with a greedy algorithm. At each step of the algorithm, an isolated taxon (future leaf) is randomly selected and inserted on a branch of the current tree such that this insertion position minimizes Fitch's score. This corresponds to the 1stRotuGbr heuristics used in [1]. Notice that each initial individual undergoes improvements using the local searcher (Section 3.4).

Algorithm 1. Hybrid Genetic and Local Search algorithm (Hydra) for the MP

input: A : a set of aligned taxa, N : the population size, M number of LS iterations
output: The most parsimonious tree found

P = *GeneratePopulation(A,N)*
for a given number of generations **do**
  $(T_1, T_2) \leftarrow$ *ChooseParents(P)*
  $T \leftarrow$ *Recombination*(T_1, T_2) *// to generate a new tree*
  $T \leftarrow$ *Local_search_improvement(T,M) // to improve an offspring*
  $P \leftarrow$ *Replace(P,T) // To insert the new offspring*
end for
return the best tree found

3.3 Fitness Evaluation

Each individual of the population is evaluated according to Definition 1 and by Fitch's algorithm. Such an evaluation assigns to the individual a parsimony score. Given that the fitness evaluation is one of the most time consuming elements of the algorithm, we use a specific implementation technique to accelerate this evaluation (see Section 3.6)

3.4 Local Search with Progressive Neighborhood

Our local searcher is based on a descent algorithm using a powerful neighborhood called *Progressive Neighborhood* (PN) introduced in [10]. Similar to the VNS (Variable Neighborhood Search) heuristic, PN modifies the size of the neighborhood during the search. But, contrary to VNS, PN starts from a medium (or large) size neighborhood like SPR or TBR [20] and progressively reduces it to the small size NNI [21] neighborhood using a parametric neighborhood relation that evolves during the search. Experiments presented in [10] showed the efficiency of this progressive neighborhood in comparison with other existing neighborhoods for phylogenetic reconstruction. The basic rational behind PN is that important tree transformations must be performed at early stages of the search using a medium or large size neighborhood (like SPR or TBR) and only minor refinements are necessary at the end of the local search process where a neighborhood like NNI is sufficient.

In order to make the neighborhood evolve, a topological distance on trees is defined in [10] that enables to build a distance matrix for a set of taxa given a tree topology. This distance is also used to control the size of the neighborhood (i.e. the distance between a pruned edge and its inserted edge is at most equal to a given distance).

Definition 5 (Topological distance). *Let i and j be two taxa of a tree T. The topological distance $\delta_T(i, j)$ between i and j is defined as the number of edges of the path between parents of i and j, minus 1 if the path contains the root of the tree.*

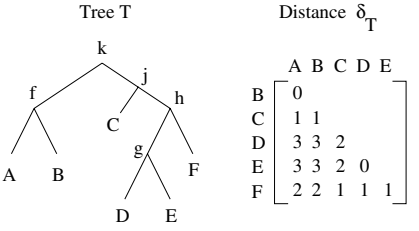


Fig. 1. Example of topological distance δ_T

For example, on figure 1, A and B have the same parent f, so $\delta_T(A, B) = 0$, and $\delta_T(A, D) = 3$ because the number of edges between f and g is 4 ($f \rightarrow k \rightarrow j \rightarrow h \rightarrow g$), and as we pass through the root node k, we decrease the value of one unit. Note that for the topological distance, we consider trees as unrooted, this is why we remove one unit when passing through the root node. The reduction process used in Hydra takes into account a parameter M which corresponds to a maximum number of LS iterations. Using this distance definition, a parameter d is introduced to control the size of the neighborhood and is defined as the distance between a pruned edge and the edge where it is reinserted (i.e. distance δ between their two descendant nodes). As such, changing d leads to neighborhoods of different sizes which are explored with a descent algorithm.

3.5 Distance-Preserving Crossover Operator DiBIPX

For phylogeny reconstruction, the individuals are trees. To create new solutions by recombination, we need a tree crossover operator. Traditionally, crossovers on phylogenetic trees follow the *subtree cutting and regrafting* strategy which consists in removing a subtree T_1' from a parent T_1 and to reinsert T_1' in the second parent T_2. Duplicated sequences (leaves) are removed from T_2. However, this process suffers from a major drawback. The topological information of trees, which are responsible of their parsimony score, is not entirely used to improve the tree. In that sense those crossovers perform a *blind* search and it would then be necessary to be able to pass in review millions of trees per second to keep up with the search efficiency.

For the purpose of creating meaningful offspring, we use the Distance-Based Information Preservation tree crossover (DiBIPX) introduced in [9]. DiBIPX is different from conventional tree crossover operators in that it explicitly takes into account the topological distance of parents trees. The key idea behind DiBIPX is the preservation of semantic information shared by the parents: if two sequences are topologically close (or far) in both parent trees, then this property should be conserved for the child. Moreover, this crossover enables to diversify the search: the child is fully rebuilt from these information. More precisely, DiBIPX is composed of three steps. First, using the topological distance (Section 3.4), the parents trees are first transformed into two distance matrices M_1 and M_2 (see Figure 1). These matrices are then combined into a new distance matrix

M_3 by a matrix operation $\oplus$. Finally, this new matrix M_3 is used to built the offspring tree using a distance-based clustering method like UPGMA [18]. In this paper, $M_3 = \alpha M_1 \oplus (1-\alpha)M_2$ such that $\forall \delta_1(i,j) \in M_1$ and $\delta_2(i,j) \in M_2$, $\delta_3(i,j) = \alpha \times min\{\delta_1(i,j), \delta_2(i,j)\} + (1-\alpha) \times max\{\delta_1(i,j), \delta_2(i,j)\}$ with $\alpha \in [0,1]$.

3.6 Acceleration of the Fitness Evaluation

During the resolution of the MP, the basic operation that is extensively used is Fitch's function (see Definition 1), which is implemented figure 2 (in C-like style). This function takes as input two taxa `t1` and `t2`. The output is the hypothetical taxon `t3` and the number of changes returned by the function. The alphabet of the DNA sequences is generally composed of 6 different symbols : $-, A, C, G, T, ?$. Where $-$ represents a gap and ? an undefined character.

```
int fitch(char t1[], char t2[], char t3[]) {
  int changes=0;
  for (int i=0;i<k;++i) {
    t3[i]= t1[i] & t2[i];
    if (t3[i]==0) { t3[i]= t1[i] | t2[i]; ++changes; }
  }
  return changes;
}
```

Fig. 2. Parsimony function traduction of Definition 1

Our implementation of this function takes full advantage of some relevant features offered by modern x86 processors. More precisely, the core of modern x86 processors has a SSE (SIMD Streaming Extension) unit which enables to vectorize the code. A SSE register is 128 bits long and can contain 16 bytes. In order to efficiently perform the union and intersection of Definition 1, each character is represented by a power of 2, from $2^0 = 1$ $(-)$ to $2^4 = 16$ (T), except for ? which can represent any other character and is then coded by the value $31 = 1 + 2 + \cdots + 16$. The union can be performed by the binary-OR (`|`) and the intersection by the binary-AND (`&`). The vectorization of Fitch's function gives a 90% improvement on Intel Core 2 Duo processors, while other architectures (pentiumII/III/4, pentium-M, Athlon 64, Sempron) provide 70 to 80 % improvement. This improvement enabled us to divide the overall computation time of our program by a factor of 3 to 4.

4 Computational Results and Comparisons

In this section, we show computational results on a set of 12 benchmark instances. To assess their quality, they are compared with the results of the very popular software TNT [11] (Tree analysis using New Technology). (TNT is perhaps today's most powerful software for phylogenetic reconstruction under PM

criterion). The code of TNT is not an open source, but is known to be highly optimized to be able to evaluate millions of trees per second. TNT integrates a large number of search strategies such as tree drifting, parsimony ratchet, sectorial search and more others. The reference software TNT was used with its default parameters specified in the documentation.

4.1 Benchmarks

A set of 12 medium to large instances (more than 100 taxa) from the TreeBase site (www.treebase.org) are used. TreeBase is a relational database of phylogenetic information which stores phylogenetic trees and the data matrices used to generate them from published research papers. For these instances, no optimum parsimony score is known.

Table 1. Parameters of the Hydra algorithm

Parameters	Values
population size	40
number of crossovers (generations)	50
local search iterations per crossover	200 000
neighborhood	progressive
α for DiBIP crossover	0.70

4.2 Parameters Tuning

In order to carry out the comparisons as fair as possible, we use for Hydra a set of standard (not fine-tuned) parameters values as follows for all the tested instances. With these parameters, each run of the algorithm undergoes $(40+50) \times 200,000 = 18 \times 10^6$ iterations (fitness evaluations with Fitch's algorithm). As we shall see later on, this number is much less than the one reached by the reference software TNT with its default parameters. According to the problem instance, the running time of one execution of Hydra goes from one to ten minutes on a Core 2 at 2.5MHz computer. Nevertheless, TNT consumes less computation time due to its extremely optimized code.

4.3 Results and Comparisons

On table 2 we report the results obtained with Hydra. For each instance, we report the following information for 20 executions: best parsimony score, maximum parsimony score, average score and standard deviation. For TNT, the software gives only two information: the best parsimony score and the total number of iterations (which corresponds to the number of evaluated candidate trees in millions).

The results show that except for one instance, Hydra obtains results of equal or better quality than TNT. Hydra reaches these results with much smaller iterations (much fewer evaluations of candidate trees) except for problems *m0972*,

Table 2. Comparison between Hydra and TNT on TreeBase instances.

problem	#taxa	length	Hydra				TNT	
			min	max	avg	std	# iter (millions)	score
m0808	178	3453	**23,777**	23,794	23,782.22	4.62	40	23,782
m0972	155	355	**1,528**	1,529	1,528.44	0.50	14	1,532
m1038	297	2021	**12,370**	12,375	12,371.22	2.19	99	12,376
m1902	209	977	**6,124**	6,129	6,126.22	1.52	44	6,131
m1987	134	618	**2,073**	2,075	2,073.67	0.92	30	**2,073**
m2055	299	2064	**2,597**	2,604	2,599.00	2.37	47	2,598
m2123	198	1426	**1,925**	1,925	1,925.00	0.00	46	**1,925**
m2725	210	8245	157,354	157,772	157,570,56	95.29	63	**157,093**
m2780	119	2020	**11,488**	11,494	11,490.78	1.90	**11**	11,490
m3275	117	5098	**21,935**	21,936	21,935.11	0.30	**9**	**21,935**
m3452	116	1157	**3,602**	3,603	3,602.33	0.49	23	**3,602**
m3453	137	995	**3,904**	3,908	3,904.78	1.42	24	**3,904**

m2780 and *m3275* for which the number of iterations of Hydra is greater than TNT. Nevertheless the experiments we have carried out show that by fixing M to 50,000 or 100,000 (i.e. **4.5 or 9 millions** iterations), we could obtain the same scores.

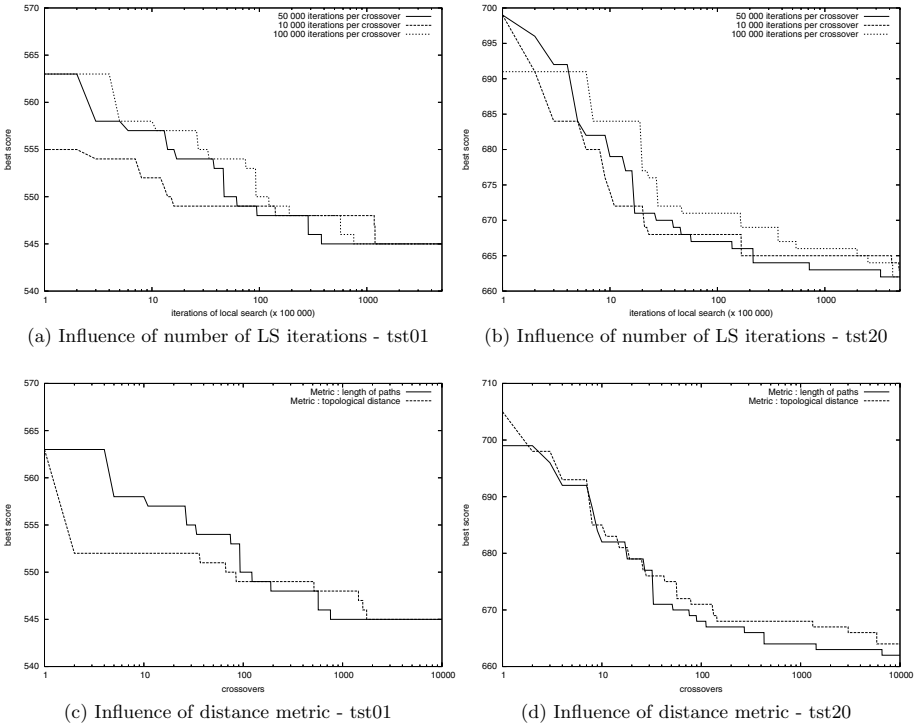
Note that the computation time of TNT is less important than Hydra because TNT implements highly specific optimization techniques to avoid a complete evaluation of the overall tree when looking for a neighbor of lower cost. Such technique is not yet implemented in Hydra.

4.4 Analysis and Discussion

In this section, we investigate and analyze the influence of some elements of the Hydra algorithm on its performance.

First, we verify how the length of the LS improvement after each crossover impacts on the convergence of the algorithm. To avoid a possible bias due to the structure of problem instance, we chose to use two *random* instances (tst01 and tst20) from [16]. We run the Hydra algorithm with three different values of local search iterations per crossover: 10 000, 50 000 and 100 000. Figure 3 (a-b) shows on a logarithmic scale, the result of the evolution of the best score averaged over 5 independent runs, each run being limited to 10^8 iterations. One observes that if the computational resources are limited, the algorithm converges faster with short local search than with long local search iterations. The influence of the local search iterations seems to decrease when the search progresses.

Second, we check the performance of DiBIPX if the topological distance defined in Section 3.4 is replaced by another distance metric. For this purpose, we use another distance which is the length of the path between two nodes of the tree taking into account the parsimony distance between each pair of neighboring nodes. Figure 3 (c-d) shows the result of the evolution of the best score



(a) Influence of number of LS iterations - tst01

(b) Influence of number of LS iterations - tst20

(c) Influence of distance metric - tst01

(d) Influence of distance metric - tst20

Fig. 3. Study of influence of local search length and distance

under the same condition as above. One observes there is no clear dominance of a distance metric over the other one, but the convergence with the topological distance is sometimes faster for a short number of LS iterations.

5 Conclusion

In this paper, we have presented the Hydra memetic algorithm for the phylogenetic tree inference problem with the Maximum Parsimony criterion. The algorithm uses a local search procedure which is based on a parametric progressive neighborhood, a specific distance-based topological tree crossover, as well as a specific implementation technique in order to accelerate the fitness evaluation.

Experimentations on a number of real benchmark instances from TreeBase show that Hydra competes very well when compared to TNT. Indeed, Hydra is able to find phylogenetic trees of better Parsimony score with much fewer evaluations of candidate trees.

However, Hydra needs more computation time than TNT. This last point constitutes an important issue to be addressed in the future. On the other hand, the techniques presented in this paper may be integrated into TNT to increase its search power.

6 Availability

Hydra is distributed with C++ source code, benchmarks and documentation and is freely available from the website of the authors. It runs under all Unix/Linux platforms. There is also a binary for Windows platforms (http://www.info.univ-angers.fr/pub/richer/rec.php).

Acknowledgements

This work was partially suported by the French Ouest Genopole® and by the BIL (Bio-Informatique Ligérienne) project (http://www.info.univ-angers.fr/bil).

References

1. Andreatta, A.A., Ribeiro, C.C.: Heuristics for the phylogeny problem. Journal of Heuristics 8, 429–447 (2002)
2. Cavalli-Sforza, L.L., Edwards, A.W.F.: Phylogenetic analysis: models and estimation procedures. Evolution 32, 550–570 (1967)
3. Edwards, A.W.F., Cavalli-Sforza, L.L.: The reconstruction of evolution. Annals of Human Genetics 27, 105–106 (1963)
4. Felsenstein, J.: Evolutionary trees from dna sequences: a maximum likelihood approach. Journal of Molecular Evolution 17, 368–376 (1981)
5. Fitch, W.: Towards defining course of evolution: minimum change for a specified tree topology. Systematic Zoology 20, 406–416 (1971)
6. Fitch, W.M., Margoliash, E.: Construction of phylogenetic trees. Science 155(3760), 279–284 (1967)
7. Foulds, L.R., Graham, R.L.: The steiner problem in phylogeny is np-complete. Advances in Applied Mathematics 3, 43–49 (1982)
8. Gascuel, O.: On the optimization principle in phylogenetic analysis and the minimum evolution criterion. Biology and Evolution 17, 401–405 (2000)
9. Goëffon, A., Richer, J.M., Hao, J.K.: A distance-based information preservation tree crossover for the maximum parsimony problem. LNCS, pp. 761–770. Springer, Heidelberg (2006)
10. Goëffon, A., Richer, J.M., Hao, J.K.: Progressive tree neighborhood applied to the maximum parsimony problem. IEEE/ACM Transactions on Computational Biology and Bioinformatics 5(1) (January-March 2008)
11. Goloboff, P.A., Farris, J.S., Nixon, K.: Tnt: Tree analysis using new technology (2003), http://www.cladistics.com/aboutTNT.html
12. Goloboff, P.A.: Character optimisation and calculation of tree lengths. Cladistics 9, 433–436 (1993)
13. Hillis, D.M., Moritz, C., Mable, B.K.: Molecular Systematics. Sinauer Associates, Inc. (1996)
14. Holland, J.H.: Adaptation in natural and artificial systems. The University of Michigan Press, Ann Arbor (1975)
15. Nixon, K.C.: The parsimony ratchet, a new method for rapid parsimony analysis. Cladistics 15, 407–414 (1999)

16. Ribeiro, C.C., Vianna, D.S.: A grasp/vnd heuristic for the phylogeny problem using a new neighborhood structure. International Transactions in Operational Research 12, 1–14 (2005)
17. Saitou, N., Nei, M.: The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol. Biol. Evol. 4, 406–425 (1987)
18. Sokal, R.R., Michener, C.D.: A statistical method for evaluating systematic relationships. University of Kansas Science Bulletin 38, 1409–1438 (1958)
19. Sokal, R.R., Sneath, P.H.A.: Principles of Numerical Taxonomy. W.H. Freeman, San Francisco (1963)
20. Swofford, D.L., Olsen, G.J.: Phylogeny Reconstruction. In: Hillis, D.M., Moritz, C. (eds.) Molecular Systematics, ch. 11, pp. 411–501 (1990)
21. Waterman, M.S., Smith, T.F.: On the similarity of dendograms. Journal of Theoretical Biology 73, 789–800 (1978)

Validation of a Morphogenesis Model of *Drosophila* Early Development by a Multi-objective Evolutionary Optimization Algorithm

Rui Dilão[1], Daniele Muraro[1], Miguel Nicolau[2], and Marc Schoenauer[2]

[1] Nonlinear Dynamics Group, IST
Department of Physics, Av. Rovisco Pais, Lisbon, Portugal
{rui,muraro}@sd.ist.utl.pt
[2] INRIA Saclay - Île-de-France
LRI- Université Paris-Sud, Paris, France
{Miguel.Nicolau,Marc.Schoenauer}@inria.fr

Abstract. We apply evolutionary computation to calibrate the parameters of a morphogenesis model of *Drosophila* early development. The model aims to describe the establishment of the steady gradients of Bicoid and Caudal proteins along the antero-posterior axis of the embryo of *Drosophila*. The model equations consist of a system of non-linear parabolic partial differential equations with initial and zero flux boundary conditions. We compare the results of single- and multi-objective variants of the CMA-ES algorithm for the model the calibration with the experimental data. Whereas the multi-objective algorithm computes a full approximation of the Pareto front, repeated runs of the single-objective algorithm give solutions that dominate (in the Pareto sense) the results of the multi-objective approach. We retain as best solutions those found by the latter technique. From the biological point of view, all such solutions are all equally acceptable, and for our test cases, the relative error between the experimental data and validated model solutions on the Pareto front are in the range $3\% - 6\%$. This technique is general and can be used as a generic tool for parameter calibration problems.

1 Introduction

The ultimate validation of mathematical or computational models of real Complex Systems can only be achieved by comparing the outcomes of the models with those of the actual system. Such models generally depend on several parameters that must be identified using experimental data. System calibration is the search for the set of parameters such that the output of the model best fits the available data. Ideally, in order to avoid possible over-fitting of the model to the data, system calibration should be performed using a data set, and the validation of the model should be done using other data sets not used during the calibration.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 176–190, 2009.

This paper deals with the validation and calibration of a reaction-diffusion model for the spatial distribution of proteins during the early stage of morphogenesis of *Drosophila*. This model incorporates the regulatory repression mechanism of Bicoid protein over *caudal* mRNA. In this case, several experimental data sets for both Bicoid and Caudal proteins are available.

Model calibration from experimental data can be formulated as an optimization problem — find the model that minimizes the difference between its outputs and the experimental data [MSM97]. Such optimization problems are usually highly multi-modal, and classical methods (e.g. gradient-based techniques) fail to give reliable solutions. Therefore, Evolutionary Algorithms (EAs) are a better choice.

In the case analyzed here, we have experimental data for the distribution of both Bicoid and Caudal proteins along the antero-posterior axis of the embryo of *Drosophila*. An ideal set of parameters for the perfect model would reach the best possible fit for both distribution, and the calibration problem could be turned into a standard optimization problem involving both fits by minimization of the sum of the Mean Square Errors (MSEs) of both models calculated with the data for both distributions. However, we are looking for meaningful biological parameters rather than for the best set of parameters. Moreover, it is likely that the orders of magnitude or the experimental errors on both proteins differ, and a simple linear aggregation of MSEs might give unbalanced results between both proteins. Therefore, a multi-objective approach seems more appropriate for our gols of model calibration and validation. On the other hand, it has been shown that the multi-objectivization of a fitness function can reduce the number of local optima [HLK08].

In order to validate these arguments, we compare the results of a single-objective approach (minimizing some weighted sums of both MSEs) to those of a multi-objective algorithm. Each trial of single-objective minimization allows us to identify one point close to the Pareto front, and hence several trials, together with good guesses for the weights of the aggregation, are necessary to sample the Pareto front reliably. The multi-objective approach lead to a full set of solutions that are hopefully close to the Pareto front. However, it turns out that for this calibration problem, a simple single-objective strategy seems to outperform better than the multi-objective approach.

This paper is structured as follows. Section 2 gives a brief description of the meaningful biological mechanisms involved in the *Drosophila* early development. Section 3 derives the reaction-diffusion model describing the production of Bicoid and Caudal from mRNAs. The parameters in the reaction-diffusion equations are to be calibrated with the experimental data. Initial conditions are given as piecewise constant functions and will be also fitted with the evolutionary algorithm. Section 4 introduces the evolutionary single- and multi-objective optimization algorithms. Section 5 applies both algorithms to the calibration of the model and compares the results. Finally, in section 6, we discuss the main conclusions of the paper.

2 Biological Background

Morphogenesis in *Drosophila* early development begins with the deposition of *bicoid* mRNA of maternal origin near one of the poles of the embryo [NV96]. During the first two hours of development, a sequence of 14 mitotic nuclear replication cycles occurs without the formation of cellular membranes around the nuclei. The early formed nuclei of the embryo lie in a single cell — the syncytial blastoderm. The nuclear membranes only appear at the end of the 14th mitotic cycle.

The absence of cellular membranes facilitates the diffusion of substances in the embryo and, during the syncytial stage, stable gradients of proteins are established. In later stages of development, the formation of the head, of the thorax and of the abdomen are associated with the patterns of distribution of proteins that took place during the previous syncytial stage of development.

After fertilization of the egg, the localized *bicoid* mRNA is translated into Bicoid protein and this protein regulates the transcription of the other zygotic genes. Other proteins of maternal origin as Caudal, Nanos or Hunchback, are produced in the early syncytial stage and are regulated by Bicoid.

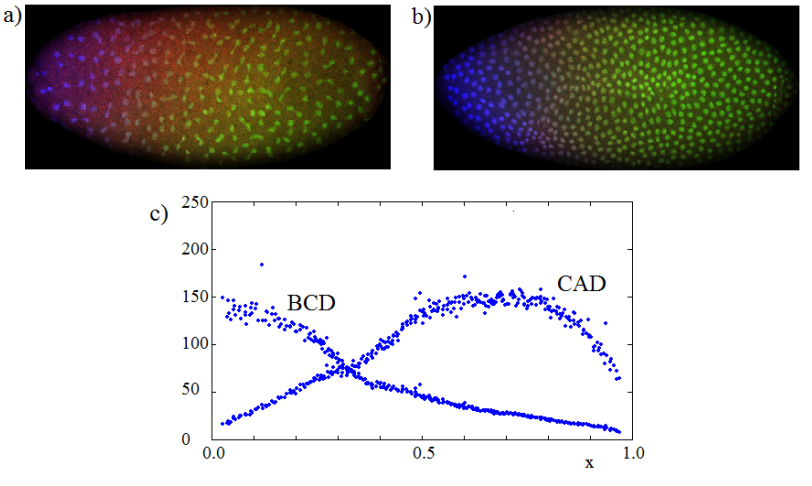


Fig. 1. Localization of Bicoid (blue) and Caudal (green) proteins near the nuclear membrane of the embryo of *Drosophila* after mitotic cycles number 11 (a) and 12 (b). This picture was taken from the datasets ab18 (a) and ab17 (b) of the FlyEx database (http://flyex.ams.sunysb.edu/flyex/, [MKRS99] and [MSKSR01]). From 1a to 1b, the nuclei have divided by mitosis, but the proteins remain sticked to the region around the nuclear membranes. In c), we show the steady state concentrations of proteins Bicoid (BCD) and Caudal (CAD) along the antero-posterior axis (x) of the embryo of *Drosophila*. This distribution has been extracted from the data set ad13 (FlyEx database), corresponding to a late stage of the mitotic cycle number 14. The horizontal axis has been scaled to the embryo length $L = 1$, and the vertical scale corresponds to light intensity arbitrary units.

During mitotic cycles 11 to 14, the observed distribution of proteins along the antero-posterior axis of *Drosophila* shows a high concentration near the nuclear membranes and a low concentration in the space between nuclei, Figure 1a and 1b.

In order to explain the observed protein gradients, a mRNA diffusion model has been proposed [DM09]. In this model, mRNA diffuses along the embryo and the produced protein stay localized near the nuclei of the mebryo. As the developmental process proceeds in time, proteins reach a steady gradient-like distribution.

Developmental processes are in general associated with the production of a cascade of regulatory processes involving genes, mRNAs and proteins. In nowadays experiments, these regulatory dependencies are specifically addressed. The simplest example is the repression effect of protein Bicoid over *caudal* mRNA, [RPJ96], Figure 1c). In this paper, we model the relationship between Bicoid and Caudal through the repression mechanism, and we further calibrate it with experimental data.

3 The Mathematical Model

During the first stage of development of *Drosophila*, the processes occurring in the embryo can be modelled in an one-dimensional domain of length L, representing the antero-posterior axis of the embryo. The *bicoid* (*bcd*) and *caudal* (*cad*) mRNA of maternal origin have initial distributions given by,

$$\begin{aligned} bcd(x, t=0) &= \begin{cases} A > 0, \text{ if } 0 < L_1 < x < L_2 < L \\ 0, \qquad \text{otherwise} \end{cases} \\ cad(x, t=0) &= \begin{cases} C > 0, \text{ if } 0 < L_3 < x < L_4 < L \\ 0, \qquad \text{otherwise} \end{cases} \end{aligned} \tag{1}$$

where L_1, L_2, L_3 and L_4 are constants defining the intervals of localization of the corresponding mRNA, and A and C are concentration constants. These distributions of mRNAs correspond to a precise initial localization as shown in experiments.

During the first stage of development, *bicoid* and *caudal* mRNAs are transformed into proteins with rate constants a_{bcd} and a_{cad}. This transformation occurs in the ribosomes, in general localized near nuclear membranes. The presence of the protein Bicoid prevents the expression of Caudal through a repression mechanism ([RPJ96]) that can be described by the mass action type transformation,

$$BCD + cad \xrightarrow{r} BCD \tag{2}$$

where r is a rate of degradation.

Introducing the hypothesis that mRNA diffuses and proteins stay localized in the embryo, and with the additional repression mechanism (2), the concentration of proteins and mRNAs in these processes evolves in time according to the model equations,

$$\begin{cases} \dfrac{\partial bcd}{\partial t} = -a_{bcd} bcd(x) + D_{bcd} \dfrac{\partial^2 bcd}{\partial x^2} \\ \dfrac{\partial BCD}{\partial t} = a_{bcd} bcd(x) \\ \dfrac{\partial cad}{\partial t} = -a_{cad} cad(x) - rBCD.cad + D_{cad} \dfrac{\partial^2 cad}{\partial x^2} \\ \dfrac{\partial CAD}{\partial t} = a_{cad} cad(x) \end{cases} \tag{3}$$

where D_{bcd} and D_{cad} are the diffusion coefficients of *bicoid* and *caudal* mRNAs, respectively. Capital letter symbols refer to protein concentrations, and lower case italic letters to mRNA concentrations.

In order to calibrate the model equations just derived in (3) with the experimental profiles, as the ones in Figure 1c), we have to identify the parameters of protein production (a_{bcd} and a_{cad}), the parameter of repression (r), the initial distribution of mRNAs (1), the ratio between the diffusion coefficients (D_{bcd}/D_{cad}, see [DJ98]), and the time, considered here as a parameter.

An ideal set of parameters calibrating an exact model with ideal experimental data would reach the best possible fit for both distributions (a zero-error fit), the calibration problem could simply be turned into a standard optimization problem involving both fits through the minimization of the sum of Mean Square Errors for both distributions. However, as in general the model is not exact, the data are noisy, and the experimental errors of both proteins might differ even by some orders of magnitude, the simple weighted sums of MSEs might give unbalanced results between both proteins.

The goal here is to find a set of parameters whose fit for both proteins has an error under a reasonable error margin, and to select an ensemble of parameters that are equivalent and well distributed inside this set of parameters. Such a goal can be rigorously defined in the setting of *multi-objective* or *Pareto optimization*.

In the following, we fit the parameters of model equations (3) for the distribution of proteins Bicoid and Caudal, with single- and multi-objective approaches. We compare the errors and computational efforts of both approaches.

4 The Algorithms

In this section, we introduce the algorithms that have been used to calibrate the model derived in the previous section. Both are based on the *Covariance Matrix Adaptation Evolution Strategy* (CMA-ES) algorithm [HO01], an Evolutionary Algorithm for black-box continuous optimization. The first algorithm is for single-objective optimization, and will be referred as CMA-ES. The second algorithm is a multi-objective version of CMA-ES, and uses embedded CMA-ES processes, together with a global Pareto-dominance based selection [IHR07].

4.1 Single-Objective Optimization: CMA-ES

Since we are dealing here with continuous optimization, the best choice of an evolutionary method is the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), first introduced in the mid-90s by Hansen and Ostermeier [HOG95, HO96], and that has reached maturity in the early 00's [HO01, HMK03, AH05]. Results from the CEC 2005 competition [Han05] and further systematic comparisons [HRM08], have shown that, thanks to its ES invariance properties, CMA-ES outperforms better than most other methods on artificial benchmarks with tunable difficulties, as well as on many real-world problems from different scientific domains.

CMA-ES is a $(\mu, \lambda)-$Evolution Strategy [Sch81], in the sense that it is an EA that uses a population of μ parents to generate λ offspring, and deterministically selects the best μ of those λ offspring for the next generation. As in all Evolution Strategies (ES), the offspring are generated by sampling a Gaussian distribution. However, in CMA-ES, this distribution is centered on a weighted recombination of the μ parents. Moreover, multidimensional Gaussian distributions are determined by their covariance matrix, a positive definite symmetrical matrix, and the art of ES lies in the way the parameters of this Gaussian mutation (i.e. the covariance matrix) are adapted on-line: CMA-ES uses the notion of *cumulated path*, i.e. modifies the matrix such that previous good moves become more likely (for further details see [HO01]). One of the important properties of CMA-ES is that it is independent of the coordinate system (*rotation-invariant*).

4.2 Evolutionary Pareto Optimization

Pareto optimality is a concept introduced by Vilfredo Pareto to evaluate the efficiency of an economic system, with applications in game theory, engineering and social sciences. It is at stake in the process of simultaneously optimizing several conflicting objectives subject to certain constraints. Pareto optimization is concerned in finding the set of *optimal trade-offs* between conflicting objectives, i.e., solutions such that the value of one objective cannot be improved without degrading the value of at least another objective. Such best compromises are what is called the *Pareto set* of the multi-objective optimization problem.

Pareto optimization is based on the notion of *dominance*. Consider a minimization problem with M real valued objective functions $f = (f_1, \ldots, f_M)$, defined on a subset $X \subset \mathbb{R}^N$, $f : X \subset \mathbb{R}^N \to \mathbb{R}^M$. A solution $x \in X$ is said to *dominate* $\bar{x} \in X$, (denoted by $x \prec \bar{x}$), if,

$$(\forall m \in \{1, \ldots, M\} : f_m(x) \leq f_m(\bar{x})) \wedge (\exists m \in \{1, \ldots, M\} : f_m(x) < f_m(\bar{x})) \ .$$

The goal of Pareto optimization is to find a good approximation to the *Pareto set*, the set of non-dominated points of the search space, generating a set of solutions in such a way that the objective values are as uniformly distributed as possible on the *Pareto front*, the image of the Pareto set in the objective space.

The classical approach of reducing the multi-objective problem into a mono-objective one by linearly aggregating all the objectives in a scalar function might

provide only a subset of the Pareto optimal solutions, as for instance it does not allow to sample the concave regions of the Pareto front [DD97]. A better idea is to use the Pareto dominance relation to select the most promising individuals within an evolutionary algorithm. Unfortunately, the dominance relation is only a partial order relation, i.e., in many cases, neither A dominates B, nor B dominates A. A secondary selection criterion is hence needed in order to get a total order relation over the search space. Many different approaches have been proposed in the last decade (see e.g. [Deb01] and [CVL02] for recent textbooks), and Evolutionary Multi-Objective Optimization is considered today a stand-alone branch of Evolutionary Algorithms with specialized workshops and conference series.

Because Covariance Matrix Adaptation was proven so successful for (single-objective) evolutionary continuous optimization, and because a multi-objective version of the algorithm has been recently proposed [IHR07], it seemed a good choice for the calibration problem at hand here.

4.3 Multi-objective CMA-ES

The Multi-Objective CMA-ES (MO-CMA-ES) [IHR07] is based on a specific (1+1)-CMA-ES algorithm, a simplified version of CMA-ES where the number of parents is set to 1, and the update of the stepsize uses a simple rule based on Rechenberg's well-known 1/5th rule [Rec73]. λ_{MO} (1+1)-CMA-ES are run in parallel, each with its own stepsize and covariance matrix. At each generation, each parent generates one offspring, and updates its mutation parameters. Then the set of λ_{MO} parents and their λ_{MO} offspring are ranked together according to the chosen selection criterion, and the best λ_{MO} carry on to the next generation.

The selection criterion goes as follows. The first sorting criterion, based on the Pareto dominance, is the non-dominated sorting proposed with NSGA-II algorithm [DPA02]: all non-dominated individuals are given rank one, and removed from the population. Amongst remaining individuals, the non-dominated ones are given rank 2, and the procedure continues until the number of required individuals is reached (λ_{MO} here). With a fast non dominated sorting approach the computational time needed to rank a population of size N can be kept of the order of MN^2 where M is the number of the objectives (see [DPA02]).

However, a second criterion is necessary, firstly in order to rank the solutions within the same rank of non-dominance, but also (and more importantly here) to guarantee a distribution as uniform as possible in the region of the objective space occupied by the Pareto front. In [IHR07] two criteria are examined: the *crowding distance* and the *contributing hypervolume*.
The crowding distance has been proposed by [DPA02] for the NSGA-II algorithm, and ranks the solutions depending on their distances to their immediate neighbors in the objective space.

Another approach is to use a metric called *S-metric* or *hypervolume measure* introduced by Zitler and Thiele [ZT98] which gives the "size of the objective value space which is covered by a set of non-dominated solutions". More precisely, it is the Lebesgue measure Λ of the union of the hypercubes a_i defined by the

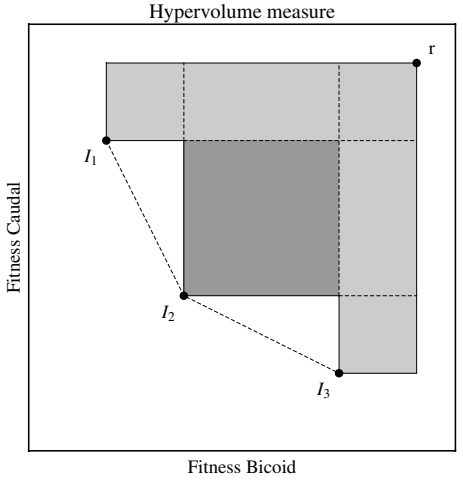


Fig. 2. The hypervolume measure is "the size of the objective value space which is covered by a set of non-dominated solutions"from a reference point (r). In light and dark grey the hypervolume measure of three non-dominated individuals I_1, I_2, I_3. In dark grey the contributing hypervolume of the individual I_2.

non-dominated points m_i and a reference point x_{ref} (see Figure 2):

$$S(\mathcal{D}) := \Lambda\Big(\Big\{\bigcup_{i=1}^{\#\mathcal{D}} a_i : m_i \in \mathcal{D}\Big\}\Big) = \Lambda\Big(\bigcup_{m\in\mathcal{D}} \{x : m \prec x \prec x_{ref}\}\Big)$$

where $\mathcal{D}$ is the set of the non-dominated points.

In [Fle03], Fleischer proved that the maximization of S constitutes a necessary and sufficient condition for the objectives to be maximally diverse Pareto optimal solutions of discrete, multi-objective, optimization problem, and proposed an algorithm to evaluate the hypervolume measure of a set in a polynomial time $O(K^3M^2)$, where K is the number of solutions in the Pareto set and M is the number of objectives. This algorithm can be efficiently implemented by an archiving strategy [KCF03]. In such a way the multi-objective problem is reduced to the single-objective one of maximizing the hypervolume measure. This measure also provides a unary indicator on the degree of success of the algorithm enabling the comparison with the results of other multi-objective algorithms.

According to [IHR07], after several tests on both criteria with two unary indicators, namely the hypervolume and the ϵ-indicator, "selection based on the hypervolume seems to be superior".

5 Experimental Calibration of the Model

5.1 The Objectives

We have applied and compared two multi-objective strategies for the calibration of the parameters in model equations (3). The experimental data has been

taken from the FlyEx database, [MKRS99] and [MSKSR01]. We have tested two algorithms on the distributions of the proteins Bicoid and Caudal along the antero-posterior axis of the embryo of *Drosophila* during cleavage cycle 14 (embryo ad13, see Figure 1). We describe these distributions by the numerical solutions of equations (3), determined by the techniques developed in [DJ98]. In order to keep the calibration as little biased as possible, we have set the number of integration steps as an unknown parameter. In this case, we are not assuming that Bicoid and Caudal protein distributions are in a steady state.

We denote by $BCD(x, \boldsymbol{\alpha})$ and $CAD(x, \boldsymbol{\alpha})$ the results of the numerical integration of equations (3), where $x \in [0, 1]$ and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m)$ is the set of parameters to be determined. The parameter search space is a hyper-rectangle in $\mathbb{R}^m$, and we denote by $\{(x_i, BCD_{exp}(x_i))\}_{i=1}^n$ and $\{(x_i, CAD_{exp}(x_i))\}_{i=1}^n$ the experimental data points. The calibration of the parameters for the model equation (3) is thus reduced to a bi-objective optimization problem, minimizing the fitness functions,

$$FitBCD(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n}(BCD(x_i, \boldsymbol{\alpha}) - BCD_{exp}(x_i))^2$$

$$FitCAD(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n}(CAD(x_i, \boldsymbol{\alpha}) - CAD_{exp}(x_i))^2$$

5.2 The Strategies

The **multi-objective strategy** uses MO-CMA-ES (Section 4.3) with a population size of $\lambda_{MO} = 100$. However, as we are not interested in the extreme parts of the Pareto Front, that have very low error value for one protein at the cost of a very high error on the other, we added a penalization to gradually eliminate large error values. More precisely, the target was to sample the Pareto front in the range $[0, 40] \times [0, 80]$ (bounds chosen after some preliminary runs), and we penalized $FitBCD$ (resp. $FitCAD$) by the amount by which $FitCAD$ (resp. $FitBCD$) overpassed its upper bound.

In total, the algorithm has been run 100 times, then the non-dominated points were extracted from the 100 populations, grouped together, leading to what can be seen as the best approximation of the Pareto Front by MO-CMA-ES.

As for the **single-objective strategy**, we used a standard aggregation technique: We have repeatedly executed CMA-ES for a family of single-objective fitness functions defined by a set of lines in the objective space with slopes c_i, namely,

$$Fit(\boldsymbol{\alpha}, c_i) = FitCAD(\boldsymbol{\alpha}) + c_i \cdot FitBCD(\boldsymbol{\alpha}), \qquad \text{where } i = 1, \ldots, 5$$

12 different slopes have been used $(0.01, 1, 5, 10, 25, 50, 60, 70, 75, 80, 90, 100)$, with 10 runs for each slope. In the end, the best results obtained for each slope has been gathered together, and the non-dominated ones are considered to be the best possible approximation of the Pareto Front for this strategy.

5.3 Results

Figure 3 presents the results of both strategies, in objective space: the approximation of the Pareto Front by MO-CMA-ES is concentrated around 8 points only, and 5 points (a-e) represent the best approximation of the Pareto Front by CMA-ES, corresponding to the slopes $(1, 5, 25, 50, 100)$. For each of these, the crosses represent the average values over the 10 runs along with standard deviations in both directions.

The first conclusion to be drawn is that MO-CMA-ES results are dominated by the CMA-ES results. Moreover, and this is the reason why so few slopes below 1 were tried, the results with slope 0.01 are slightly dominated, but very close to those with slope 1 (hence they are not plotted on Figure 3). This seems to indicate that $FitCAD$ has orders of magnitude more influence on the fits than $FitBCD$. In order to confirm this point, we ran CMA-ES on each fitness alone: it reaches error values down to $FitCAD = 47$, but at the cost of an error on Bicoid of order 10^6 - while $FitBCD$ never reached any value lower than 28.

Another finding was that the final populations of the MO-CMA-ES runs (not shown) were very diverse, whereas all single-objective runs for the same slope robustly found very similar solutions, as witnessed by the crosses on Figure 3. MO-CMA-ES seems to lack some evolutionary pressure toward the true Pareto Front, maybe because, in multi-objective algorithms, the whole population rapidly contains only non-dominated points, and the selection pressure is then

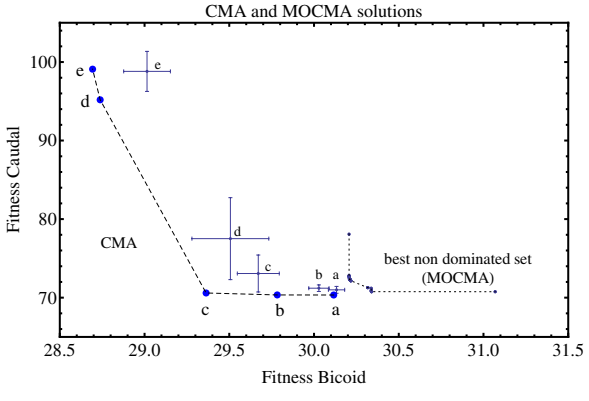


Fig. 3. Best non-dominated sets found by the single-objective (CMA) and multi-objective (MO-CMA) techniques. In the single-objective approach, we have optimized the fitness function $FitCAD + c_i \cdot FitBCD$, for $c_i = 1, 5, 25, 50, 100$. By construction, the Pareto front is tangent to these lines. The results are labelled as small a, b, c, d, e, respectively. For each slope c_i, the mean values and the standard deviations are represented with crosses. The best result of each set of 10 optimization runs for each c_i has the same label. The final result of the multi-objective case is the best non-dominated population calculated by the intersection of the final populations of 100 independent runs. In this case, the MO-CMA technique gives worst solutions for the optimization problem.

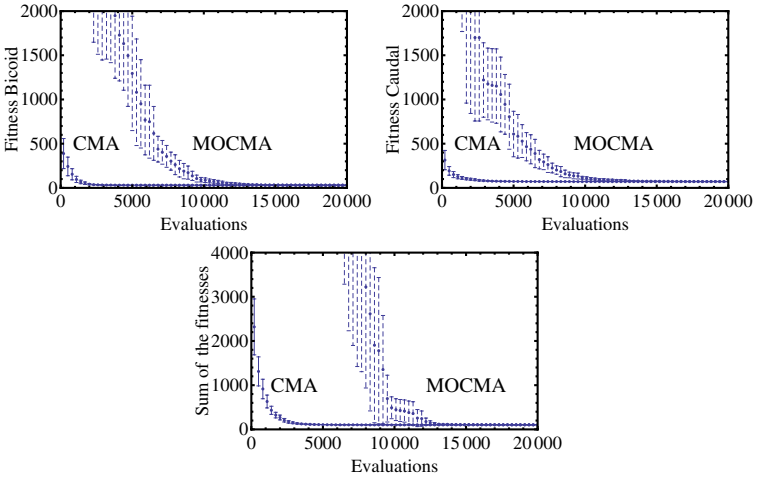


Fig. 4. Evolution of the fitnesses (MSEs on the Bicoid and Caudal) during optimization. Plotted are the averages and standard deviations over 100 runs for both CMA-ES, minimizing the sum of both fitnesses, and MO-CMA-ES. Above are single fitness values (the minimal values in the population for MO-CMA-ES, the values for the best individual in CMA-ES). Below is the sum of both of the above (the actual fitness used by CMA-ES).

enforcing the uniform spreading on the current front rather than progress toward the true Pareto Front.

Figure 4 illustrates the situation, by plotting average results for 100 runs of MO-CMA-ES on the one hand, and of CMA-ES optimizing $FitCAD(\boldsymbol{\alpha}) + FitBCD(\boldsymbol{\alpha})$ (i.e. corresponding to point a in figure 3) on the other hand. Above plots are the best values in the population (averaged over the 100 runs) for $FitCAD$ and $FitBCD$, and below is the sum of both. The fact that the MO-CMA-ES plot never catches up that of CMA-ES, even when considering the best possible value for one error alone, is another sign of its poor behavior.

Note that both algorithms are compared using the number of function evaluations on the x-axis, but the MO-CMA-ES requires some additional overhead time: the update of the covariance matrix is made for each individual, and the whole population undergoes non-dominated sorting. However, for larger systems of differential equations, such as the ones describing the genetic network of the early development of *Drosophila*, [AD06], the computational cost of numerical integration will be the most time consuming part of the algorithm. This is why it has also been used here.

Table 1 displays the parameters of model equations (3) and the corresponding fitness values, fitted with the CMA-ES algorithm. The actual fits of the corresponding solutions of the model equations (3) and plotted against experimental data are shown in Figure 5. In the numerical fits, we have fixed the *Drosophila*

Table 1. Parameter values for the five best non-dominated solutions of model equations (3), obtained with the CMA algorithm, for the experimental data set of Figure 1c). In Figure 5, we show this data set together with the solutions of equations (3) for the parameter values a-e. All the different choices of these parameter values are calibrated candidates of the experimental data set. We also show, for each parameter, the mean value (mean) and the standard deviation (σ) taken on the Pareto front.

	a	b	c	d	e	mean	σ
L_1	$5.68 \cdot 10^{-2}$	$6.72 \cdot 10^{-2}$	$6.25 \cdot 10^{-2}$	$3.29 \cdot 10^{-2}$	$1.43 \cdot 10^{-2}$	$4.67 \cdot 10^{-2}$	$2.24 \cdot 10^{-2}$
L_2	$1.73 \cdot 10^{-1}$	$1.68 \cdot 10^{-1}$	$1.62 \cdot 10^{-1}$	$1.84 \cdot 10^{-1}$	$1.94 \cdot 10^{-1}$	$1.76 \cdot 10^{-1}$	$0.12 \cdot 10^{-1}$
L_3	$4.28 \cdot 10^{-1}$	$4.35 \cdot 10^{-1}$	$4.04 \cdot 10^{-1}$	$4.07 \cdot 10^{-1}$	$4.04 \cdot 10^{-1}$	$4.16 \cdot 10^{-1}$	$0.14 \cdot 10^{-1}$
L_4	$7.63 \cdot 10^{-1}$	$7.74 \cdot 10^{-1}$	$8.45 \cdot 10^{-1}$	$8.45 \cdot 10^{-1}$	$8.48 \cdot 10^{-1}$	$8.15 \cdot 10^{-1}$	$0.42 \cdot 10^{-1}$
B	$1.53 \cdot 10^{+3}$	$1.98 \cdot 10^{+3}$	$3.47 \cdot 10^{+3}$	$2.36 \cdot 10^{+3}$	$1.98 \cdot 10^{+3}$	$2.26 \cdot 10^{+3}$	$0.73 \cdot 10^{+3}$
C	$1.06 \cdot 10^{+3}$	$1.08 \cdot 10^{+3}$	$1.26 \cdot 10^{+3}$	$1.28 \cdot 10^{+3}$	$1.28 \cdot 10^{+3}$	$1.19 \cdot 10^{+3}$	$0.11 \cdot 10^{+3}$
D_{bcd}	$1.00 \cdot 10^{-2}$	$1.09 \cdot 10^{-2}$	$1.99 \cdot 10^{-2}$	$2.03 \cdot 10^{-2}$	$2.04 \cdot 10^{-2}$	$1.63 \cdot 10^{-2}$	$0.53 \cdot 10^{-2}$
D_{cad}	$1.00 \cdot 10^{-2}$	$1.00 \cdot 10^{-2}$	$1.00 \cdot 10^{-2}$	$1.00 \cdot 10^{-2}$	$1.00 \cdot 10^{-2}$	$1.00 \cdot 10^{-2}$	$0.00 \cdot 10^{-2}$
a_{bcd}	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$1.31 \cdot 10^{+1}$
a_{cad}	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$9.99 \cdot 10^{+4}$	$3.96 \cdot 10^{+1}$
r	$8.64 \cdot 10^{+3}$	$6.74 \cdot 10^{+3}$	$3.34 \cdot 10^{-2}$	$5.74 \cdot 10^{-2}$	$6.71 \cdot 10^{-4}$	$3.07 \cdot 10^{+3}$	$4.26 \cdot 10^{+3}$
Iterations	$9.84 \cdot 10^{+3}$	$9.79 \cdot 10^{+3}$	$9.37 \cdot 10^{+3}$	$9.35 \cdot 10^{+3}$	$9.36 \cdot 10^{+3}$	$9.54 \cdot 10^{+3}$	$0.25 \cdot 10^{+3}$

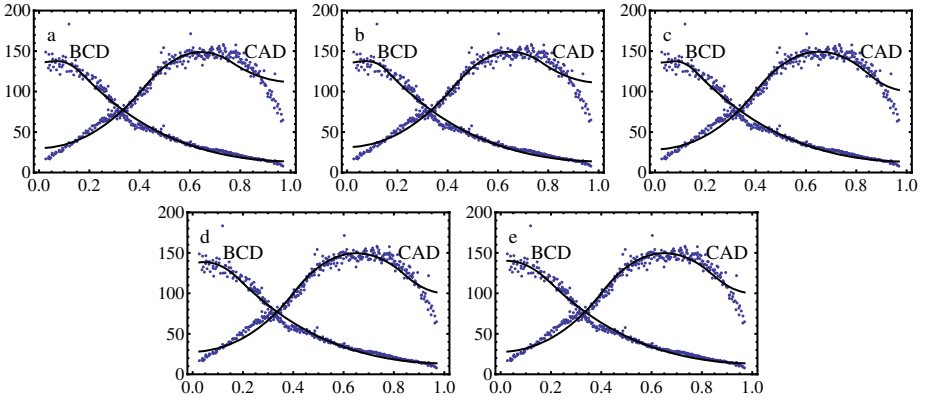


Fig. 5. Fits with experimental data for Bicoid and Caudal from the data shown in Figure 1c). (Embryo ad13 from the FlyEx database). The five figures correspond to the parameter values a-e in Table 1. The parameters of the different functions are on the best approximated Pareto set of Figure 3. In (a), we show the best fit for Bicoid and the worst fit for Caudal, then a gradual variation occurs, and in (e) it we have plotted the worst fit for Bicoid and the best fit for Caudal.

embryo length to the standard value $L = 0.5 \times 10^{-3} m$, and all the graphs of Figure 5 have been scaled to the interval $[0, 1]$.

The mean relative error between the experimental data and the optimized solutions of the model equations (3) can be measured by the fitness function.

For example, for the case of Bicoid protein, if BCD_{max} is the maximum value of the experimental values, the relative error of the calibrated model equations can be measured by $\sqrt{FitBCD/BCD^2_{max}}$. For the experimental data analyzed here, the mean relative error of all solution in the approximated Pareto front using the single-objective strategy are in the range $3\% - 6\%$.

6 Discussion and Conclusion

We have tested the applicability of a single- and a multi-objective algorithms to the calibration of a model equation for a biological process. We have used two approaches: reducing the multi-objective optimization problem to a parametrized single-objective problem, repeatedly tackled by CMA-ES algorithm; and use an *ab-initio* multi-objective perspective, the multi-objective version of CMA-ES.

From a Computer Science perspective, the most striking fact is the difference in performance between both algorithms. Further experiments with other multi-objective algorithms should be run before any conclusion can be drawn. However, in this paper, the multi-objective approach lacks pressure toward the Pareto front, suggesting that other algorithms with a better control of the convergence to the Pareto front should be tried. Another important issue that must be tested is the generalization issue, i.e. how well the parameters that have been identified using one experimental dataset fit another dataset gathered from the same biological system.

From the biological point of view, we have shown that, if multi-objectives are considered, biological data is compatible with a large set of parameters values associated with a specific model. This non-dominated variability, intrinsic to biological systems, can explain the phenotypic plasticity of living systems.

On the other hand, from a more practical point of view, this problem enabled us to show the applicability of an mRNA diffusion model in order to describe the establishment of steady gradients of proteins in *Drosophila* early development.

Acknowledgements

This work was entirely funded by European project GENNETEC (FP6 STREP IST 034952). We also want to heartily thank Nikolaus Hansen for the fruitful discussions we had with him about this work, and his efficient advices.

References

[AD06] Alves, F., Dilão, R.: Modelling segmental patterning in Drosophila: Maternal and gap genes. Journal of Theoretical Biology 241, 342–359 (2006)

[CVL02] Coello Coello, C.A., Van Veldhuizen, D.A., Lamont, G.B.: Evolutionary algorithms for solving multi-objective problems. Kluwer Academic Publishers, Dordrecht

[DD97] Das, I., Dennis, J.E.: A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. Structural optimization 14(1), 63–69 (1997)

[Deb01] Deb, K.: Multi-objective optimization using evolutionary algorithms. Wiley, Chichester (2001)

[DPA02] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2) (2002)

[DM09] Dilão, R., Muraro, D.: Calibration and validation of mathematical models describing the gradient of Bicoid in the embryo of *Drosophila* (2009) (preprint)

[DJ98] Dilão, R., Sainhas, J.: Validation and Calibration of Models for Reaction-Diffusion Systems. Int. J. of Bifurcation and Chaos 8, 1163–1182 (1998)

[EBN05] Emmerich, M., Beume, N., Naujoks, B.: An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 62–76. Springer, Heidelberg (2005)

[Fle03] Fleischer, M.: The Measure of Pareto Optima. Applications to Multi-objective Metaheuristics. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 519–533. Springer, Heidelberg (2003)

[HLK08] Handl, J., Lovell, S.C., Knowles, J.: Investigations into the Effect of Multiobjectivisation in Protein Structure Prediction. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N., et al. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 702–711. Springer, Heidelberg (2008)

[HOG95] Hansen, N., Ostermeier, A., Gawelczyk, A.: On the Adaptation of Arbitrary Normal Mutation Distributions in Evolution Strategies: The Generating Set Adaptation. In: ICGA 1995, pp. 57–64 (1995)

[HO96] Hansen, N., Ostermeier, A.: Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In: ICEC 1996, pp. 312–317. IEEE Press, Los Alamitos (1996)

[HO01] Hansen, N., Ostermeier, A.: Completely Derandomized Self-Adaptation in Evolution Strategies. Evolutionary Computation 9(2), 159–195 (2001)

[HMK03] Hansen, N., Müller, S., Koumoutsakos, P.: Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). Evolutionary Computation 11(1) (2003)

[AH05] Auger, A., Hansen, N.: A Restart CMA Evolution Strategy With Increasing Population Size. In: CEC 2005, pp. 1769–1776. IEEE Press, Los Alamitos (2005)

[Han05] Hansen, N., et al.: Comparison of Evolutionary Algorithms on a Benchmark Function Set. In: CEC 2005 Special Session (2005)

[HRM08] Hansen, N., Ros, R., Mauny, N., Schoenauer, M., Auger, A.: PSO Facing Non-Separable and Ill-Conditioned Problems. INRIA Research Report number RR-6447 (2008)

[IHR07] Igel, C., Hansen, N., Roth, S.: Covariance Matrix Adaptation for Multi-objective Optimization. Evolutionary Computation 15(1), 1–28 (2007)

[KCF03] Knowles, J.D., Corne, D.W., Fleisher, M.: Bounded Archiving using the Lebesgue Measure. In: Proceedings of CEC 2003, vol. 4, pp. 2490–2497. IEEE Press, Los Alamitos (2003)

[MKRS99] Myasnikova, E., Kosman, D., Reinitz, J., Samsonova, M.: Spatio-temporal registration of the expression patterns of *Drosophila* segmentation genes. In: Seventh International Conference on Intelligent Systems for Molecular Biology, pp. 195–201. AAAI Press, Menlo Park (1999)

[MSKSR01] Myasnikova, E., Samsonova, A., Kozlov, K., Samsonova, M., Reinitz, J.: Registration of the expression patterns of *Drosophila* segmentation genes by two independent methods. Bioinformatics 17(1), 3–12 (2001)

[NV96] Nusslein-Volhard, C.: Gradients that organize embryo development. Scientific American, 54–61 (1996)

[Rec73] Rechenberg, I.: Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution, Werkstatt Bionik und Evolutionstechnik. Frommann-Holzboog, Stuttgart (1973)

[RPJ96] Rivera-Pomar, R., Jäckle, H.: From gradients to stripes in Drosophila embryogenesis: Filling in the gaps. Trends Genet. 12, 478–483 (1996)

[Sch81] Schwefel, H.-P.: Numerical Optimization of Computer Models. J. Wiley & Sons, New York (1981–1995)

[MSM97] Sebag, M., Schoenauer, M., Maitournam, H.: Parametric and non-parametric identification of macro-mechanical models. In: Quadraglia, D., et al. (eds.) Genetic Algorithms and Evolution Strategies in Engineering and Computer Sciences, pp. 327–340. John Wiley, Chichester (1997)

[ZT98] Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms - a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 292–301. Springer, Heidelberg (1998)

Refining Genetic Algorithm Based Fuzzy Clustering through Supervised Learning for Unsupervised Cancer Classification

Anirban Mukhopadhyay[1], Ujjwal Maulik[2], and Sanghamitra Bandyopadhyay[3]

[1] Department of Computer Science and Engineering, University of Kalyani, Kalyani-741235, India
anirban@klyuniv.ac.in
[2] Department of Computer Science and Engineering, Jadavpur University, Kolkata-700032, India
drumaulik@cse.jdvu.ac.in
[3] Machine Intelligence Unit, Indian Statistical Institute, Kolkata-700108, India
sanghami@isical.ac.in

Abstract. Fuzzy clustering is an important tool for analyzing microarray cancer data sets in order classify the tissue samples. This article describes a real-coded Genetic Algorithm (GA) based fuzzy clustering method that combines with popular Artificial Neural Network (ANN) / Support vector Machine (SVM) based classifier in this purpose. The clustering produced by GA is refined using ANN / SVM classifier to obtain improved clustering performance. The proposed technique is used to cluster three publicly available real life microarray cancer data sets. The performance of the proposed clustering method has been compared to several other microarray clustering algorithms for three publicly available benchmark cancer data sets, viz., leukemia, Colon cancer and Lymphoma data to establish its superiority.

1 Introduction

With the advancement of microarray technology, it is now possible to measure the expression levels of a huge number of genes across different tissue samples simultaneously [1,2,3,4,5]. Several studies have been done in the area of supervised cancer classification. But unsupervised classification or clustering of tissue samples should also be studied since in many cases, labeled tissue samples are not available. This article explores the application of the GA based fuzzy clustering for unsupervised classification of cancer data.

A microarray gene expression data consisting of g genes and s tissue samples is typically organized in a 2D matrix $E = [e_{ij}]$ of size $s \times g$. Each element e_{ij} gives the expression level of the jth gene for the ith tissue sample. Clustering [6] is a popular unsupervised pattern classification technique which partitions the input space into K regions $\{C_1, C_2, \ldots, C_K\}$ based on some similarity/dissimilarity metric where the value of K may or may not be known *a priori*. The main objective of any clustering technique is to produce a $K \times n$ partition matrix

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2009, LNCS 5483, pp. 191–202, 2009.

$U(X)$ of the given data set X, consisting of n patterns, $X = \{x_1, x_2, \ldots, x_n\}$. The partition matrix may be represented as $U = [u_{kj}]$, $k = 1, \ldots, K$ and $j = 1, \ldots, n$, where u_{kj} is the membership of pattern x_j to cluster C_k. In crisp partitioning $u_{kj} = 1$ if $x_j \in C_k$, otherwise $u_{kj} = 0$. On the other hand, for fuzzy partitioning of the data, the following conditions hold on U (representing non-degenerate clustering): $0 < \sum_{j=1}^{n} u_{kj} < n$, $\sum_{k=1}^{K} u_{kj} = 1$, and $\sum_{k=1}^{K} \sum_{j=1}^{n} u_{kj} = n$.

For defuzzification of a fuzzy clustering solution, samples are assigned to clusters to which they have the highest membership degree. It has been observed that for a particular cluster, some of the samples belonging to it have higher membership degree to that cluster, whereas the other samples of the same cluster may have lower membership degree. Thus the samples in the later case are not assigned to that cluster with high confidence. Motivated by this, the clustering result produced by the fuzzy clustering technique is refined using Artificial Neural Network (ANN) [7,8] / Support vector Machine (SVM) [9,10], which is trained by the points with high membership degree in a cluster. The trained ANN / SVM classifier can thereafter be used to classify the remaining points. A real-coded Genetic Algorithm (GA) based fuzzy clustering algorithm has been used for generating the fuzzy partition matrix. In the subsequent stage, ANN / SVM is applied to classify the points with lower membership degree.

The performance of the proposed GA-ANN and GA-SVM clustering techniques has been demonstrated on three publicly available benchmark cancer data sets, viz., Leukemia, Colon cancer, and Lymphoma data and compared with that of GA based clustering alone, K-means clustering [6], FCM algorithm [11], hierarchical average linkage clustering [6] and Self Organizing Map (SOM) clustering [12].

2 GA Based Fuzzy Clustering

Genetic Algorithms (GAs) [13] are randomized search and optimization techniques guided by the principles of evolution and natural genetics, and have a large amount of implicit parallelism. They provide near-optimal solutions of an objective or fitness in complex, large and multimodal landscapes. In GAs, the parameters of the search space are encoded in the form of strings (or, chromosomes). A fitness is associated with each string that represents the degree of goodness of the solution encoded in it. Biologically inspired operators like selection, crossover and mutation are used over a number of generations for generating potentially better strings. Genetic and other evolutionary algorithms have been earlier used for pattern classification including clustering of data [5,14,15]. In this section, an improved GA based fuzzy clustering algorithm [14] is described.

2.1 Chromosome Encoding and Initial Population

Here the chromosomes are made up of real numbers which represent the coordinates of the cluster centers. If chromosome i encodes the centers of K clusters in g dimensional space then its length l_i will be $g \times K$. For example, in four

dimensional space, the chromosome

<1.3 11.4 53.8 2.6 10.1 21.4 0.4 5.3 35.6 0.0 10.3 17.6>

encodes 3 cluster centers, (1.3, 11.4, 53.8, 2.6), (10.1, 21.4, 0.4, 5.3) and (35.6, 0.0, 10.3, 17.6). Each center is considered to be indivisible.

Each chromosome in the initial population consists of the coordinates of K random points from the data set.

2.2 Computation of Fitness

The fitness of a chromosome indicates the degree of goodness of the solution it represents. J_m cluster validity index [11] is used as a fitness here. Let $X = \{x_1, x_2, \ldots, x_n\}$ be the set of n data points to be clustered. For computing the fitness, the centers encoded in a chromosome are first extracted. Let these be denoted as $Z = \{z_1, z_2, \ldots, z_K\}$. The membership values u_{ik}, $i = 1, 2, \ldots, K$ and $k = 1, 2, \ldots, n$ are computed as per Eqn. 1 [11]:

$$u_{ik} = \frac{1}{\sum_{j=1}^{K} (\frac{D(z_i, x_k)}{D(z_j, x_k)})^{\frac{2}{m-1}}}, 1 \leq i \leq K; \quad 1 \leq k \leq n, \tag{1}$$

where $D(.,.)$ is a distance function, m is the weighting coefficient and K be the number of clusters encoded in the chromosome. (Note that while computing u_{ik} using Eqn. 1, if $D(z_j, x_k)$ is equal to zero for some j, then u_{ik} is set to zero for all $i = 1, \ldots, K$, $i \neq j$, while u_{jk} is set equal to one.) Subsequently, the centers encoded in a chromosome are updated using Eqn. 2 [11]:

$$z_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m}, \quad 1 \leq i \leq K, \tag{2}$$

and the cluster membership values are recomputed as per Eqn. 1. The objective function is the J_m validity measure that is optimized by the FCM algorithm. This computes the global fuzzy variance of the clusters and this is expressed by the following equation [11]:

$$J_m = \sum_{k=1}^{n} \sum_{i=1}^{K} u_{ik}^m D^2(z_i, x_k), \tag{3}$$

J_m is to be minimized to get compact clusters.

2.3 Selection

To generate the mating pool of chromosomes, conventional proportional selection based on roulette wheel technique [13] has been used. Here, a string receives a number of copies proportional to its fitness in the mating pool.

2.4 Crossover

Conventional single point crossover for variable length chromosomes is used here. While choosing the crossover points, the cluster centers are considered to be indivisible. Hence the crossover points can only lie in between two clusters centers. The crossover operator is applied stochastically with probability p_c.

2.5 Mutation

Following mutation operator is adopted depending on the mutation probability p_m. In this method, a random center of the chromosome to be mutated is chosen. Then a random number δ in the range [0, 1] is generated with uniform distribution. If the value of the center in the dth dimension at is z_d, after mutation it becomes $(1 \pm 2.\delta).z_d$, when $z_d \neq 0$, and $(\pm 2.\delta)$, when $z_d = 0$. The '+' or '-' sign occurs with equal probability.

2.6 Elitism

Elitism is required to track the best chromosome obtained till the most recent generation. It is implemented as follows: If the fitness of the best chromosome of the ith generation is better than the fitness of the worst chromosome of the $(i + 1)$th generation, then the worst chromosome of the $(i + 1)$th generation is replaced by the best chromosome of the ith generation.

2.7 Termination Condition

The algorithm has been executed for a fixed number of generations. The population size is kept constant throughout all the generations. The best string of the last generation is considered as the solution given by the algorithm.

3 Artificial Neural Network Based Classifier

The ANN classifier algorithm implements a three layer feed-forward neural network with a hyperbolic tangent function for the hidden layer and the softmax function [16] for the output layer. Using softmax, output of ith output neuron is given by:

$$p_i = \frac{e^{q_i}}{\sum_{j=1}^{K} e^{q_j}}, \tag{4}$$

where q_i the net input to the ith output neuron, and K is the number of output neurons. The use of softmax makes it possible to interpret the outputs as probabilities. The number of neurons in the input layer is d, where d is the number of features of the input data set. The number of neurons in the output layer is K, where K is the number of classes. The ith output neuron provides the class membership degree of the input pattern to the ith class. The number of hidden layer neurons is taken as $2 \times d$. The weights are optimized with a maximum a posteriori (MAP) approach; cross-entropy error function augmented with a Gaussian prior over the weights. The regularization is determined by MacKay's ML-II scheme [8]. Outlier probability of training examples is also estimated [17].

4 Support Vector Machines

Viewing the input data as two sets of vectors in a d-dimensional space, a Support Vector Machine (SVM) classifier [9] constructs a maximally separating hyperplane to separate the two classes of points in that space. On each side of the separating hyperplane, two parallel hyperplanes are constructed that are pushed up against the two classes of points. A good separation is achieved by the hyperplane that has the largest distance to the neighboring data points of both the classes. Larger distance between these parallel hyperplanes indicates better generalization error of the classifier. Fundamentally the SVM classifier is designed for two-class problems. It can be extended for multi-class problems by designing a number of two-class SVMs.

Suppose a data set contains n feature vectors $< x_i, y_i >$, where $y_i \in \{+1, -1\}$, denotes the class label for the data point x_i. The problem of finding the weight vector w can be formulated as minimizing the following function:

$$L(w) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i, \tag{5}$$

subject to $y_i[w.\phi(x_i) + b] \geq 1 - \xi_i, \quad i = 1, \ldots, n, \quad \xi_i \geq 0$. Here, b is the bias and the function $\phi(x)$ maps the input vector to the feature vector. The dual formulation is given by maximizing the following:

$$Q(\lambda) = \sum_{i=1}^{n}\lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \lambda_i \lambda_j \kappa(x_i, x_j), \tag{6}$$

subject to $\sum_{i=1}^{n} y_i\lambda_i = 0$ and $0 \leq \lambda_i \leq C, \quad i = 1, \ldots, n$. The parameter C, called as regularization parameter, controls the tradeoff between complexity of the SVM and the misclassification rate. Only a small fraction of the λ_i coefficients are nonzero. The corresponding pairs of x_i entries are known as support vectors and they fully define the decision function. Geometrically, the support vectors are the points lying near the separating hyperplane. $\kappa(x_i, x_j) = \phi(x_i).\phi(x_j)$ is the *kernel function*.

Kernel functions map the input space into higher dimensional space. Linear, polynomial, sigmoidal, radial basis function (RBF), etc. are examples of them. RBF kernels are of the following form:

$$\kappa(x_i, x_j) = e^{-\gamma|x_i - x_j|^2}, \tag{7}$$

where γ is the weight. The above mentioned RBF kernel is used here and $\gamma = 0.9$ is taken.

5 Combining GA Based Fuzzy Clustering with ANN and SVM Classifier

In this section we have described how the GA based fuzzy clustering is combined with ANN and SVM classifier for performance improvement. The steps of GA-ANN / GA-SVM clustering are as follows:

1. **GA based Clustering:** Cluster the input data set $X = \{x_1, x2, \ldots, x_n\}$ using GA based fuzzy clustering algorithm to evolve the fuzzy membership matrix $U = [u_{ik}]$, $i = 1, \ldots, K$ and $k = 1, \ldots, n$, where K and n be the number of clusters and the number of data points, respectively.

2. **Defuzzification:** Assign each point k, $(k = 1, \ldots, n)$, to some cluster j $(1 \leq j \leq K)$ such that $u_{jk} = \max_{i=1,\ldots,K}\{u_{ik}\}$.

3. **Selecting Training Points:** For each cluster i $(i = 1, \ldots, K)$, sort the points assigned to it in the descending order of their membership degrees to that cluster and then select top P% points from the sorted list as the training points for that cluster. Combine the training points of all the clusters to form the complete training set. Keep the remaining points as the test set.

4. **Training of the Classifier:** Train the ANN / SVM classifier using the training set created in the previous step.

5. **Classify Test Points:** Predict the class labels of the remaining points (test points) using the trained ANN / SVM classifier.

6. **Final Clustering:** Combine the training set (class labels given by GA-clustering) and the test set (class labels given by ANN / SVM) to obtain the complete label vector and return it as the clustering solution.

The membership threshold P has been varied from 20% to 80% with step size 5% and J_m index value is computed for each value of P. The value of P, for which the minimum J_m index score is obtained is taken as the optimum threshold and the corresponding clustering solution is returned. Note that the above process involves in running GA only once, whereas training and testing involving ANN / SVM classifier need to be executed for every value of P.

6 Data Sets and Pre-processing

Three publicly available benchmark cancer data sets, viz., Leukemia, Colon cancer and Lymphoma data sets have been used for experiments. the data sets and their pre-processing are described in this section.

6.1 Leukemia Data

The Leukemia data set [1] consists of 72 tissue samples. The samples consists of two types of leukemia, 25 of AML and 48 of ALL. The samples are taken from 63 bone marrow samples and 9 peripheral blood samples. There are 7,129 genes in the data set. The data set is available at http://www.genome.wi.mit.edu/MPR.

The data set is subjected to a number of pre-processing steps to find out the genes with most variability. As here we consider the problem of unsupervised

classification, the gene selection steps followed here are also completely unsupervised. However, more sophisticated methods for gene selection could have been applied. First we select the genes whose expression levels fall between 100 and 15000. From the resulting 1015 genes, the 100 genes with the largest variation across samples are selected, and the remaining expression values are log-transformed. The resultant data set is of dimension 72×100.

6.2 Colon Cancer Data

The Colon cancer data set [4] consists of 62 samples of colon epithelial cells from colon cancer patients. The samples consists of tumor biopsies collected from tumors (40 samples), and normal biopsies collected from healthy part of the colons (22 samples) of the same patient. The number of genes in the data set is 2000. The data set is publicly available at http://microarray.princeton.edu/oncology.

This data set is pre-processed as follows: First the genes whose expression levels fall between 10 and 15000 are selected. From the resulting 1765 genes, the 200 genes with the largest variation across samples are selected, and the remaining expression values are log-transformed. The resultant data set is of dimension 62×200.

6.3 Lymphoma Data

The diffuse large B-cell lymphoma (DLBCL) dataset [2] contains expression measurements of 96 normal and malignant lymphocyte samples each measured using a specialized cDNA microarray, containing 4,026 genes that are preferentially expressed in lymphoid cells or which are of known immunological or oncological importance. There are 42 DLBCL and 54 other cancer disease samples. The data set is publicly available at http://genome-www.stanford.edu/lymphoma.

The pre-processing steps for this data sets are as follows: As the data set contains some missing values, we select only those genes which do not contain any missing value. This results in 748 genes. Next each gene is normalized to have expression value between 0 and 1. Thereafter top 100 genes with respect to variance are selected. Hence the data set contains 96 samples each described by 100 genes.

7 Experimental Results

The performance of the proposed GA-ANN and GA-SVM clustering has been compared with GA based clustering alone, K-means clustering [6], FCM [11], hierarchical average linkage clustering method [6] and Self Organizing MAP (SOM) clustering [12].

7.1 Performance Metrics

For evaluating the performance of the clustering algorithms on the three cancer data sets, an external validity measure namely Adjusted Rand Index (ARI) [18] and an internal validity measure namely Silhouette Index ($\mathcal{S}(C)$) [19] are used.

Adjusted Rand Index. Suppose T is the true clustering of the samples of a cancer data set based on domain knowledge and C a clustering result given by some clustering algorithm. Let a, b, c and d respectively denote the number of sample pairs belonging to the same cluster in both T and C, the number of pairs belonging to the same cluster in T but to different clusters in C, the number of pairs belonging to different clusters in T but to the same cluster in C, and the number of pairs belonging to different clusters in both T and C. The adjusted Rand index $ARI(T, C)$ is then defined as follows:

$$ARI(T, C) = \frac{2(ad - bc)}{(a + b)(b + d) + (a + c)(c + d)}. \tag{8}$$

The value of $ARI(T, C)$ lies between 0 and 1 and higher value indicates that C is more similar to T. Also, $ARI(T, T) = 1$.

Silhouette Index. Silhouette index [19] is a cluster validity index that is used to judge the quality of any clustering solution C. Suppose a represents the average distance of a point from the other points of the cluster to which the point is assigned, and b represents the minimum of the average distances of the point from the points of the other clusters. Now the silhouette width s of the point is defined as:

$$s = \frac{b - a}{max\{a, b\}}. \tag{9}$$

Silhouette index $\mathcal{S}(C)$ is the average silhouette width of all the data points (genes) and it reflects the compactness and separation of clusters. The value of silhouette index varies from -1 to 1 and higher value indicates better clustering result.

7.2 Distance Measure and Input Parameters

Pearson correlation based distance measure is used here since it is independent of the expression values, rather it is dependent on the expression patterns of the two samples between which the distance is to be measured. Hence the data sets need not to be standardized with zero mean and unit variance for using the correlation based distance measures. This is defined below:

Pearson Correlation: Given two sample vectors, s_i and s_j, Pearson correlation coefficient $Cor(s_i, s_j)$ between them is computed as:

$$Cor(s_i, s_j) = \frac{\sum_{l=1}^{p}(s_{il} - \mu_{s_i})(s_{jl} - \mu_{s_j})}{\sqrt{\sum_{l=1}^{p}(s_{il} - \mu_{s_i})^2}\sqrt{\sum_{l=1}^{p}(s_{jl} - \mu_{s_j})^2}}. \tag{10}$$

Here μ_{s_i} and μ_{s_j} represent the arithmetic means of the components of the sample vectors s_i and s_j respectively. Pearson correlation coefficient defined in Eqn. 10 is a measure of similarity between two samples in the feature space. The distance between two samples s_i and s_j is computed as $1 - Cor(s_i, s_j)$, which represents the dissimilarity between those two samples.

The different parameters of GA are taken as follows: number of generations=100, population size=50, crossover probability=0.8 and mutation probability=0.01. These values have been chosen experimentally. The fuzzy exponent m is chosen as in [20,21], and the values of m for the data sets Leukemia, Colon cancer and Lymphoma are set to 1.19, 1.53 and 1.34, respectively. The K-means and the fuzzy C-means algorithms have been run for 200 iterations unless they converge before that.

7.3 Results

Each algorithm has been executed for 50 runs and the Tables 1 and 2 report the average ARI index scores and average $S(C)$ index scores over 50 runs, respectively, for the Leukemia, Colon cancer and Lymphoma data sets. It is evident from the tables that irrespective of the data set used, application of ANN / SVM improves the clustering performance of the GA based clustering. The GA-ANN / GA-SVM clustering produce the best average ARI index scores and $S(C)$ index scores compared to the other algorithms. In general, it can be noted that

Table 1. Average ARI scores produced by 50 runs of different algorithms for Leukemia, Colon cancer and Lymphoma data sets

Algorithms	Leukemia	Colon cancer	Lymphoma
GA-ANN	0.8324	0.6634	0.4015
GA-SVM	0.8133	0.6672	0.4031
K-means	0.7093	0.4264	0.3017
FCM	0.7019	0.4794	0.3414
GA	0.7758	0.5629	0.3458
Avg. linkage	0.6284	-0.0432	0.3973
SOM	0.7409	0.5923	0.3367

Table 2. Average $S(C)$ scores produced by 50 runs of different algorithms for Leukemia, Colon cancer and Lymphoma data sets

Algorithms	Leukemia	Colon cancer	Lymphoma
GA-ANN	0.3108	0.4338	0.3226
GA-SVM	0.3102	0.4385	0.3234
K-means	0.2252	0.1344	0.2291
FCM	0.2453	0.1649	0.2239
GA	0.2606	0.3015	0.2543
Avg. linkage	0.1934	0.2653	0.2235
SOM	0.2518	0.3153	0.2665

GA-ANN / GA-SVM consistently outperforms the other algorithms for all the data sets in terms of both the performance indices.

7.4 Statistical Significance Test

To establish that GA-ANN and GA-SVM are significantly superior compared to the other algorithms, a statistical significance test called t-test has been conducted at the 5% significance level. Seven groups, corresponding to the seven algorithms (1. GA-ANN, 2. GA-SVM, 3. K-means, 4. FCM, 5. GA, 6. average linkage and 7. SOM) have been created for each data set. Each group consists of the *ARI* index scores produced by 50 runs of the corresponding algorithm.

As is evident from the Table 1, the average values of *ARI* scores for GA-ANN and GA-SVM are better than those for the other algorithms. To establish that this goodness is statistically significant, Table 3 reports the *P-values* produced by t-test for comparison of two groups (group corresponding to GA-ANN and a group corresponding to some other algorithm) at a time. As a null hypothesis, it is assumed that there are no significant difference between the mean values of two groups. Whereas, the alternative hypothesis is that there is significant difference in the mean values of the two groups. All the *P-values*, except for comparison between GA-ANN and GA-SVM, reported in the table are less than 0.05 (5% significance level). Similar results are obtained for GA-SVM also (Table 4). These are strong evidences against the null hypothesis, indicating that the better mean values of the *ARI* index produced by GA-ANN and GA-SVM are statistically significant and do not occur by chance. Moreover, large *P-values* for comparison between GA-ANN and GA-SVM indicates that these two algorithms are equally good. Statistical significance test on $S(C)$ index values (not shown here) also provides similar results.

Table 3. *P-values* produced by t-test comparing GA-ANN with other algorithms

Data Sets	P-values (comparing average *ARI* index scores of GA-ANN with other algorithms)					
	K-means	FCM	GA	Avg. Linkage	SOM	GA-SVM
Leukemia	2.3E-06	2.67E-06	3.4E-05	1.08E-06	2.5E-04	0.537
Colon	2.31E-05	4.43E-08	4.3E-07	6.5E-05	1.45E-06	0.621
Lymphoma	4.42E-06	4.61E-07	4.2E-05	1.4E-05	1.08E-06	0.844

Table 4. *P-values* produced by t-test comparing GA-SVM with other algorithms

Data Sets	P-values (comparing average *ARI* index scores of GA-SVM with other algorithms)					
	K-means	FCM	GA	Avg. Linkage	SOM	GA-ANN
Leukemia	1.3E-06	2.17E-06	3.5E-06	1.14E-05	5.25E-04	0.537
Colon	2.31E-05	2.23E-08	1.3E-04	5.52E-04	2.31E-05	0.621
Lymphoma	5.52E-06	7.4E-05	2.4E-05	2.43E-06	3.7E-06	0.844

8 Discussion and Conclusions

An unsupervised cancer data classification technique based on GA-based fuzzy clustering has been presented. The quality of the clustering solutions produced by the GA-based fuzzy clustering has been improved through ANN / SVM classification. Results on three publicly available benchmark cancer data sets, viz., Leukemia, Colon cancer and Lymphoma, have been demonstrated. The performance of the proposed technique has been compared with that of several other clustering methods. The results demonstrate how improvement in clustering performance is achieved by refining the clustering solution produced by GA using ANN / SVM classifier. It has been found that the GA-ANN / GA-SVM clustering schemes outperform all the other clustering methods considered here.

Acknowledgement

Sanghamitra Bandyopadhyay gratefully acknowledges the financial support from the grant no. DST/SJF/ET-02/2006-07 under the Swarnajayanti Fellowship scheme of the Department of Science and Technology, Government of India.

References

1. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gassenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomeld, D.D., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
2. Alizadeh, A.A., Eisen, M.B., Davis, R., Ma, C., Lossos, I., Rosenwald, A., Boldrick, J., Warnke, R., Levy, R., Wilson, W., Grever, M., Byrd, J., Botstein, D., Brown, P.O., Straudt, L.M.: Distinct types of diffuse large b-cell lymphomas identified by gene expression profiling. Nature 403, 503–511 (2000)
3. Yeung, K.Y., Bumgarner, R.E.: Multiclass classification of microarray data with repeated measurements: application to cancer. Genome Biology 4 (2003)
4. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proceedings of National Academy of Science, Cell Biology 96, 6745–6750 (1999)
5. Bandyopadhyay, S., Mukhopadhyay, A., Maulik, U.: An improved algorithm for clustering gene expression data. Bioinformatics 23(21), 2859–2865 (2007)
6. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
7. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press, Oxford (1996)
8. MacKay, D.J.C.: The evidence framework applied to classification networks. Neural Computation 4(5), 720–736 (1992)
9. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
10. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. J. Machine Learning Research 2, 265–292 (2001)

11. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981)
12. Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E., Golub, T.: Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. Proc. Nat. Academy of Sciences 96, 2907–2912 (1999)
13. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, New York (1989)
14. Maulik, U., Bandyopadhyay, S.: Genetic algorithm based clustering technique. Pattern Recognition 33, 1455–1465 (2000)
15. Maulik, U., Bandyopadhyay, S.: Fuzzy partitioning using a real-coded variable-length genetic algorithm for pixel classification. IEEE Transactions on Geoscience and Remote Sensing 41(5), 1075–1081 (2003)
16. Andersen, L.N., Larsen, J., Hansen, L.K., HintzMadsen, M.: Adaptive regularization of neural classifiers. In: Proc. IEEE Workshop on Neural Networks for Signal Processing VII, New York, USA, pp. 24–33 (1997)
17. Sigurdsson, S., Larsen, J., Hansen, L.: Outlier estimation and detection: Application to skin lesion classification. In: Proc. Intl. Conf. on Acoustics, Speech and Signal Processing (2002)
18. Yeung, K.Y., Ruzzo, W.L.: An empirical study on principal component analysis for clustering gene expression data. Bioinformatics 17(9), 763–774 (2001)
19. Rousseeuw, P.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comp. App. Math. 20, 53–65 (1987)
20. Kim, S.Y., Lee, J.W., Bae, J.S.: Effect of data normalization on fuzzy clustering of dna microarray data. BMC Bioinformatics 7(134) (2006)
21. Dembele, D., Kastner, P.: Fuzzy c-means method for clustering microarray data. Bioinformatics 19(8), 973–980 (2003)

Author Index