STUDIES IN FUZZINESS AND SOFT COMPUTING

Yaochu Jin
Lipo Wang (Eds.)

Fuzzy Systems in Bioinformatics and Computational Biology

Springer

Studies in Fuzziness and Soft Computing, Volume 242

Further volumes of this series can be found on our homepage: springer.com

Yaochu Jin and Lipo Wang

Fuzzy Systems in Bioinformatics and Computational Biology

Springer

Editors

Dr. Yaochu Jin
Honda Research Institute
Europe GmbH
Carl-Legien-Str. 30
63073 Offenbach
Germany
E-Mail: yaochu.jin@honda-ri.de

Prof. Dr. Lipo Wang
Nanyang Technological University
School of Electrical & Electronic
Engineering
50 Nanyang Avenue
Singapore 639798
E-Mail: elpwang@ntu.edu.sg

ISBN 978-3-540-89967-9 e-ISBN 978-3-540-89968-6

DOI 10.1007/978-3-540-89968-6

Studies in Fuzziness and Soft Computing ISSN 1434-9922

Library of Congress Control Number: 2008942029

Typeset & Cover Design: Scientific Publishing Services Pvt. Ltd., Chennai, India.

Printed in acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

To our families

Preface

Biological systems are inherently stochastic and uncertain. Thus, research in bioinformatics, where computer technologies are applied to the management of biological data, and in computational biology, where computational models are built for modeling and analysis of ecological, molecular, cellular and neural networks, has to deal with a large amount of uncertainties. For example, a small number of molecules, different internal states of a population of cells, changes in environments and genetic mutations can all contribute to variations in gene expression.

Fuzzy logic has shown to be a powerful tool in capturing different uncertainties in engineering systems. In recent years, fuzzy logic based modeling and analysis approaches are becoming popular in analyzing biological data and modeling biological systems. Numerous results have been reported to demonstrate the effectiveness in applying fuzzy logic to solving a wide range of problems in bioinformatics, biomedical engineering, and computational biology.

This book contains 16 chapters that represent a body of selected research work on applying fuzzy systems to the modeling and analysis of biological systems, in particular, to bioinformatics, biomedical engineering and computational biology.

In Chapter 1, a method for generating type-1 and type-2 fuzzy rules using artificial immune systems (AIS) is presented. A brief introduction to both AIS and type-2 fuzzy systems is provided. The potential application of AIS-based fuzzy systems to to data mining in bioinformatics and biomedecine is discussed.

Chapter 2 describes a framework for performing assembly of genome sequences of both single and multiple organisms using fuzzy logic. It is shown that fuzzy logic improves the performance of genome sequence assembly by allowing for tolerance of inexactness or errors in fragment matching and enhances the classification of fragments belonging to different organisms with a divide-and-conquer strategy.

Chapter 3 develops a model-based approach to the analysis of proteobacterial genomes for promoter features that is able to account for the variability in sequence, location and topology intrinsic to differential gene expression. Authors decompose a feature into a family of models or building blocks, which maximizes

the sensitivity of detecting those instances that weakly resemble a consensus (e.g., binding site sequences) without decreasing the specificity. These features are treated using fuzzy assignments, which allow them to encode how well a particular sequence matches each of the multiple models for a given promoter feature.

A data-adaptive fuzzy filtering framework for processing of cDNA microarray images is presented in Chapter 4. The framework is designed to remove noise in cDNA microarray images that does not require for fuzzy rules, nor does it assume that the original cDNA signal is available. This is achieved by utilizing the inference engine in the form of transformed distance metrics between the cDNA vector-valued samples within the supporting window. In this way, the training or learning of the weighting coefficients is only based on local image features without the use of linguistic fuzzy rules or local statistics estimation.

In Chapter 5, the fuzzy c-means clustering algorithm is suggested for analyzing microarray gene expression data followed by a discussion of the main concerns in clustering gene expression data. Tuning of the parameters are discussed in the context of 2-way and 3-way microarray data. A transformation that allows for more contrast in distances between all pairs of samples in a dataset is proposed, which increases the likelihood of detection of a group structure in a high dimensional dataset.

Chapter 6 introduces a flexible framework for feature selection and classification of microarray data. Dimensionality reduction is achieved by the application of a supervised fuzzy pattern algorithm that is able to reduce and discretize existing gene expression profiles. Then, a self-organizing neural network, termed growing cell structures (GCS) network, is employed for clustering biological data.

In Chapter 7, authors employ a fuzzy rule-based classification system to analyze gene expression data. The applied classifier consists of a set of fuzzy if-then rules that allow for efficient and accurate classification of input patterns. Furthermore, a hybrid fuzzy approach to classifying gene expression data, where a genetic algorithm is used to select a subset of the fuzzy rules, is also presented. It is shown that the performance of the compact fuzzy classifier is comparable to that of the full classifier.

Reconstruction of gene regulatory networks using fuzzy logic based models is reviewed in both Chapters 8 and 9. Chapter 8 emphasizes the functionalities of regulatory motifs, and the reconstruction of such motifs using fuzzy systems. Two selected methods are discussed in detail, examples are given where the two methods are applied to both real microarray data concerning the yeast cell cycle and simulated data concerning the Raf signaling pathway. In contrast, Chapter 9 surveys the application of fuzzy logic both to clustering of gene expression data and the reconstruction of gene regulatory networks. Examples are also supplied in the discussion of the methods.

Chapters 10, 11 and 12 provides examples of employing fuzzy logic to model gene expression data and biological networks in greater detail. Chapter 8 describes the use of genetic programming to evolve a fuzzy rule base to model gene expression. It is shown that fuzzy rule based models allow for the insertion of

prior knowledge, which makes it possible to find sets of rules that include the relationships between genes that are already known. In addition, it is demonstrated that evolving a fuzzy rule base using genetic programming is able to extract explanatory rules from microarray data obtained in the real experiments.

Chapter 11 describes a class of widely used neuro-fuzzy systems, known as adaptive neuro-fuzzy inference systems (ANFIS), and its application to modeling gene regulatory networks. Furthermore, a hierarchical, multi-layer ANFIS model (termed GeneCFE-ANFIS) is introduced. It is shown that GeneCFE-ANFIS is able to improve the performance of prediction in terms of true positive rate with a little amount of a priori knowledge about gene interactions.

Although fuzzy logic presents an appealing approach to modeling gene expression data, it also faces the serious challenge of combinatorial rule explosion in modeling complex biological networks. Chapter 12 suggests a number of approaches to addressing the scalability issue, including adopting the union rule configuration or optimizing the fuzzy rule structure using genetic algorithms.

Chapters 13, 14 and 15 provide various applications of fuzzy logic to dealing with biomedical problems. In Chapter 13, authors provide an overview of several fuzzy c-means based clustering approaches to medical imaging. The conventional hard c-means and the fuzzy c-means, together with three computationally more efficient variants of fuzzy c-means are evaluated. In Chapter 14, the application of self-organizing fuzzy logic controller (SOFLC) to the control of a multivariable model of anesthesia is explored. A methodology is proposed to design SOFLC for complex multi-input/multi-output (MIMO) systems. Different design strategies of MIMO are outlined and the application of such SOFLC systems to muscle relaxation and depth of anesthesia control is studied. Chapter 15 presents an interval type-2 fuzzy classifier and its application to ECG arrhythmic classification problem. It is shown that the uncertainties associated with the membership functions can be encapsulated by the footprint of uncertainty (FOU) and that it can be fully characterized by the upper membership function (UMF) and lower membership function (LMF). The proposed type-2 fuzzy classifier is applied to the ECG arrhythmic classification problem and the performance of the classifier is tested on MIT-BIH Arrhythmia database. Results show that the proposed strategies to design the FOU are essential to achieve a high performance fuzzy rule-based classifier in the presence of uncertainties.

The role of fuzzy logic in the modeling of gene regulatory networks is further studied in Chapter 16. Different to its applications in modeling and analysis of gene expression data, this chapter investigates in silico the influence of control logic on the easiness of evolving typical regulatory dynamics in computational models of genetic regulatory networks. The gene regulatory network motif considered in this work consists of three genes with both positive and negative feedback loops. Two fuzzy logic formulations are studied in this work, one is known as the Zadeh operator, and other is the probabilistic operator. Empirical results show that with the probabilistic 'AND' operator and the probabilistic 'OR' operator, the system is able to evolve sustained oscillation with a low probability. However, sustained oscillation is not evolvable when the Zadeh operator is

employed. In addition, it is also shown that regulatory motifs with the probabilistic operators possess much richer dynamics than that with the Zadeh operators.

The research work described in this book presents a selected yet comprehensive picture of how fuzzy logic can elegantly address problems in bioinformatics, biomedecical engineering and computational biology, particularly in dealing with uncertainties in biological systems. We hope that the methodologies and application examples discussed in the book are instructive and inspiring to both practitioners and researchers. Thus, we hope that the publication of this book will further promote the related research areas.

We would like to thank Dr. Janusz Kacprzyk for including this book in the Springer book series "Studies in Fuzziness and Soft Computing". We are also grateful to the authors for their nice contributions and cooperation during the preparation of the book. Finally, we would like to thank Heather King and Thomas Ditzinger of Springer for their kind support and patience.

October 2008

Yaochu Jin
Lipo Wang

Contents

List of Contributors

M.F. Abbod
School of Engineering and Design
Brunel University
West London, United Kingdom
Maysam.Abbod@brunel.ac.uk

Vitoantonio Bevilacqua
Polytechnic of Bari
Via E. Orabona 4
70125 Bari, Italy

Amit Bhaya
COPPE-UFRJ
amit@nacad.ufrj.br

Adrienne Breland
Department of Computer Science & Engineering
University of Nevada
Reno, Reno NV 89557, USA
brelanda@cse.unr.edu

Guy N. Brock
University of Louisville
Louisville, KY, USA
guy.brock@louisville.edu

Chung-Ming Chen
Institute of Biomedical Engineering
National Taiwan University
Taipei, 106, Taiwan

Luonan Chen
Institute of Systems Biology
Shanghai University, Shanghai
200444, China
Department of Electrical Engineering and Electronics
Osaka Sangyo University
Osaka 574-8530, Japan

Teck Wee Chua
Department of Electrical Engineering
National University of Singapore
Singapore
cteckwee@nus.edu.sg

Cheng-Long Chuang
Institute of Biomedical Engineering
National Taiwan University
Taipei, 106, Taiwan
clchuang@ieee.org
Department of Bio-Industrial Mechatronics Engineering
National Taiwan University
Taipei, 106, Taiwan

Juan M. Corchado
Department of Computer Science
University of Salamanca
Salamanca 37008, Spain
corchado@usal.es

Suman Datta
School of Biomedical Engineering
Science & Health Systems
Drexel University 3141 Chestnut St.
Philadelphia, PA 19104 USA
sumandatta82@gmail.com

Doulaye Dembélé
IGBMC, CNRS-INSERM-ULP
1 rue Laurent Fries, BP 10142
67404 Illkirch Cedex, France
doulaye@titus.u-strasbg.fr

S.Z. Fan
Department of Anesthesiology
College of Medicine
National Taiwan University
Taipei, Taiwan.

Fernando Díaz
Department of Computer Science
University of Valladolid
Segovia 40005, Spain
fdiaz@infor.uva.es

Florentino Fdez-Riverola
Department of Computer Science
University of Vigo
Ourense 32004, Spain
riverola@uvigo.es

Daniel Glez-Peña
Department of Computer Science
University of Vigo
Ourense, 32004, Spain
dgpena@uvigo.es

Y.Y. Han
Department of Trauma
National Taiwan University Hospital
Taipei, Taiwan

Oscar Harari
Dept. Computer Science and
Artificial Intelligence
University of Granada
E-18071, Spain
oharari@decsai.ugr.es

Frederick C. Harris Jr.
Department of Computer Science &
Engineering
University of Nevada
Reno, Reno NV 89557, USA
Fred.Harris@cse.unr.edu

Luis Herrera
Dept. Computer Science and
Artificial Intelligence
University of Granada
E-18071, Spain

C.Y. Hsu
Department of Electrical Engineering
Yuan Ze University, Taiwan

Xiaohua Hu
College of Information Science and
Technology
Drexel University
thu@cis.ist.drexel.edu

S.J. Huang
Department of Surgery
National Taiwan University Hospital
Taipei, Taiwan

Hisao Ishibuchi
Department of Computer Science and
Intelligent Systems
Osaka Prefecture University
Osaka, Japan
hisaoi@cs.osakafu-u.ac.jp

Joe-Air Jiang
Department of Bio-Industrial
Mechatronics Engineering
National Taiwan University
Taipei, 106, Taiwan

Yaochu Jin
Honda Research Institute Europe
Carl-Legien-Str. 30
63073 Offenbach, Germany
yaochu.jin@honda-ri.de

Laura Kubatko
The Ohio State University
Columbus, OH, USA
lkubatko@stat.osu.edu

Ricardo Linden
Faculdade Salesiana Maria
Auxiliadora CEPEL
rlinden@pobox.com

Rastislav Lukac
Epson Edge
3771 Victoria Park Avenue
Toronto, Ontario, M1W 3Z5
Canada
lukacr@ieee.org

Giuseppe Mastronardi
Polytechnic of Bari
Via E. Orabona 4
70125 Bari, Italy

José R. Méndez
Department of Computer Science
University of Vigo
Ourense, 32004, Spain
moncho.mendez@uvigo.es

Filippo Menolascina
Polytechnic of Bari
Via E. Orabona 4
70125 Bari, Italy
f.menolascina@ieee.org

Tomoharu Nakashima
Department of Computer Science
and Intelligent Systems
Osaka Prefecture University
Osaka, Japan
nakashi@cs.osakafu-u.ac.jp

Sara Nasser
Department of Computer Science &
Engineering
University of Nevada
Reno, Reno NV 89557, USA
sara@cse.unr.edu

Monica Nicolescu
Department of Computer Science &
Engineering
University of Nevada
Reno, Reno NV 89557, USA
monica@cse.unr.edu

Vasyl Pihur
University of Louisville
Louisville, KY, USA

Konstantinos N. Plataniotis
The Edward S. Rogers Sr.
Department of ECE
University of Toronto,
10 King's College Road
Toronto, Ontario, M5S 3G4
Canada

Gerald Schaefer
School of Engineering and Applied
Science
Aston University
Birmingham, U.K.
G.Schaefer@aston.ac.uk

Bernhard Sendhoff
Honda Research Institute Europe
Carl-Legien-Str. 30
63073 Offenbach, Germany
bernhard.sendhoff@honda-ri.de

Chunmei Shi
People's Hospital of Guangxi
Nanning, China

J.S. Shieh
Department of Mechanical Engineering
Yuan Ze University
Taiwan
jsshieh@saturn.yzu.edu.tw

Bahrad A. Sokhansanj
School of Biomedical Engineering Science & Health Systems
Drexel University 3141 Chestnut St.
Philadelphia, PA 19104, USA
bahrad.sokhansanj@drexel.edu

Woei Wan Tan
Department of Electrical Engineering
National University of Singapore
Singapore
wwtan@nus.edu.sg

Gregory L. Vert
Department of Computer Science & Engineering
University of Nevada
Reno, Reno NV 89557, USA
gvert@cse.unr.edu

Rui-Sheng Wang
School of Information
Renmin University of China
Beijing 100872, China
Department of Electrical Engineering and Electronics
Osaka Sangyo University
Osaka 574-8530, Japan

Mariadele Zarrilli
Polytechnic of Bari
Via E. Orabona 4
70125 Bari, Italy
f.menolascina@ieee.org

Shihua Zhang
Academy of Mathematics and Systems Science
Chinese Academy of Sciences
Beijing 100080, China
Graduate School
Chinese Academy of Sciences
Beijing 100049, China
zsh@amss.ac.cn

Huiyu Zhou
School of Engineering and Design
Brunel University
Uxbridge, U.K.
Huiyu.Zhou@brunel.ac.uk

Igor Zwir
Howard Hughes Medical Institute
Department of Molecular Microbiology
Washington University School of Medicine
St. Louis, MO 63110-1093, USA
zwir@borcimwustl.edu
Dept. Computer Science and Artificial Intelligence
University of Granada
E-18071, Spain

Shihua Zhang
Academy of Mathematics and Systems Science
Chinese Academy of Sciences
Beijing 100080, China
Graduate School
Chinese Academy of Sciences
Beijing 100049, China
zsh@amss.ac.cn

Xiang-Sun Zhang
Academy of Mathematics and Systems Science
Chinese Academy of Sciences
Beijing 100080, China

1

Induction of Fuzzy Rules by Means of Artificial Immune Systems in Bioinformatics

Filippo Menolascina, Vitoantonio Bevilacqua, Mariadele Zarrilli, and Giuseppe Mastronardi

Polytechnic of Bari, Via E. Orabona 4, 70125 Bari, Italy
f.menolascina@ieee.org

Summary. Fuzzy Rule Induction (FRI) is one of the main areas of research in the field of computational intelligence. Recently FRI has been successfully employed in the field of data mining in bioinformatics [34, 38]. Thanks to its flexibility and potentialities FRI allowed researchers to extract rules that can be easily modeled in natural language and submitted to experts in the field that can validate their accuracy or consistency. The process of FRI can result to be highly complex from a computational complexity point of view and, for this reason, several alternative approaches to accomplish this process have been proposed ranging from iterative and simultaneous algorithms [22] to Genetic Algorithms and Ant Colony Optimization based approaches [22]. In this chapter we will focus on a specific application of type-1 (T1) and type-2(T2) fuzzy systems to data mining in bioinformatics in which FRI is carried out using a novel and promising computational paradigm, namely Artificial Immune Systems (AIS). In order to provide the reader with the necessary theoretical background we will go through a brief introduction to the fields of AIS and T2 Fuzzy Systems, then we will set up the scientific context and describe applications of these concepts to real world cases. Conclusions and cues for future work in this fascinating field will be provided in the end.

1.1 Artificial Immune Systems

Artificial Immune Systems (AIS) represent one of the most recent and promising approaches in the branch of bio-inspired techniques. Although this open field of research is still in its infancy, several relevant results have been achieved by using the AIS paradigm in demanding tasks such as the those coming from computational biology and biochemistry. Artificial immune systems (AIS) can be defined as computational systems inspired by theoretical immunology, observed immune functions, principles and mechanisms in order to solve problems. Their development and application domains follow those of soft computing paradigms such as artificial neural networks (ANN), evolutionary algorithms (EA) and fuzzy systems (FS). Soft computing was the term coined to address a new trend of coexistence and integration that reflects a high degree of interaction among several computational intelligence approaches ike artificial neural network, evolutionary algorithms and fuzzy systems. The idea of integrating different computational

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 1–17.
springerlink.com

intelligence paradigms in order to create hybrids combining the strengths of different approaches is not new. Following the previous concepts when in 2002 de Castro and Timmis introduced AIS as a new soft computing paradigm they gave birth to a new challenge to have a great potential to interact the new born technique with others. Strictly speaking evolution and immune system are biologically closely related to each other. In fact the process of natural selection can be seen to act the immune system at two levels. First recall that lymphocytes multiply based on their affinity with a pathogen. The higher affinity lymphocytes are selected to reproduce, a process usually named immune microevolution. The mechanism of immune microevolution is very important. The clonal selection principle presupposes that a very large number of *B-cells* containing antigenic receptors is constantly circulating throughout the organism. The great diversity of this repertoire is a result of the random genetic recombination of gene fragments from different libraries plus the random insertion of gene sequences during cell development. This availability of different solutions guarantees that at least one cell will produce an antibody capable of recognizing, thus binding with, any antigen that invades the organism. The antigen-antibody binding stimulates the production of clones of the selected cells, where successive generations result in exponential growth of the selected antibody type. Some of these antibodies remain in circulation even after the immune response ceases, constituting a sort of immune memory. Other cells differentiate in plasma cells, producing antibodies in high rates. Finally during reproduction, some clones suffer an affinity maturation process, where somatic mutations are inserted with high rates (hypermutation) and, combined with a strong selective mechanism, improve the capability (*Ag-Ab* affinity and clone size) of these antibodies to recognize and respond to the selective antigens. Secondly, there is surely an immune contribution to natural selection, which acts by allowing the multiplication of those people carrying genes that are most able to provide maximal defense against infectious diseases coupled with minimal risk of autoimmune diseases. At this time the majority of the immune algorithms currently developed have an evolutionary type of learning of embodied process and several techniques from one strategy have been used to enhance another.

The success of the AIS paradigm is based on two key properties of its theoretical foundations: recognition and adaptation/optimization. When an animal is exposed to an antigen, some subpopulation of its bone marrow derived cells (*B lymphocytes*) respond by producing antibodies (*Ab*). Each cell secretes a single type of antibody, which is relatively specific for the antigen. By binding to these antibodies (*cell receptors*), and with a second signal from accessory cells, such as the T-helper cell, the antigen stimulates the *B cell* to proliferate (divide) and mature into terminal (non-dividing) antibody secreting cells, called plasma cells. The process of cell division (mitosis) generates a clone, i.e., a cell or set of cells that are the progenies of a single cell. While plasma cells are the most active antibody secretors, large B lymphocytes, which divide rapidly, also secrete antibodies, albeit at a lower rate. On the other hand, T cells play a central role in the regulation of the *B cell* response and are preeminent in cell

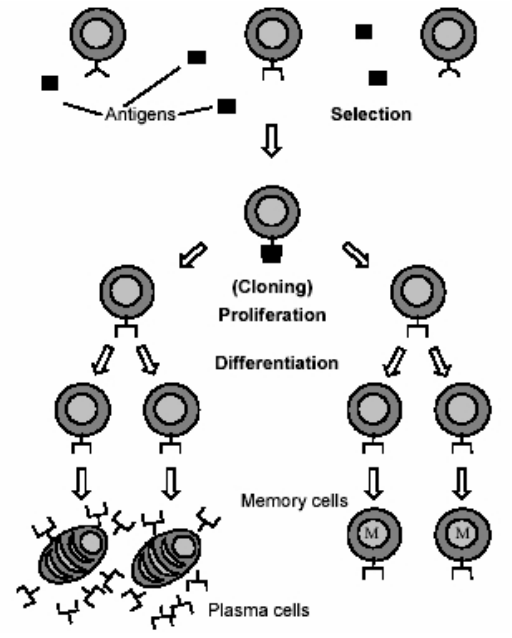


Fig. 1.1. Clonal selection principle in natural immune systems

mediated immune responses, but will not be explicitly accounted for the development of our model. Lymphocytes, in addition to proliferating and/or differentiating into plasma cells, can differentiate into long-lived B memory cells. Memory cells circulate through the blood, lymph and tissues, and when exposed to a second antigenic stimulus commence to differentiate into large lymphocytes capable of producing high affinity antibodies, pre-selected for the specific antigen that had stimulated the primary response. Fig 1.1 depicts the clonal selection principle.

The clonal selection and affinity maturation principles are used to explain how the immune system reacts to pathogens and how it improves its capability of recognizing and eliminating pathogens [14]. In a simple form, clonal selection states that when a pathogen invades the organism, a number of immune cells that recognize these pathogens will proliferate; some of them will become effector cells, while others will be maintained as memory cells. The effector cells secrete antibodies in large numbers, and the memory cells have long life spans so as to act faster and more effectively in future exposures to the same or a similar pathogen. During the cellular reproduction, the cells suffer somatic mutations with high rates and, together with a selective force, the higher affinity cells in relation to the invading pathogen differentiate into memory cells. This whole process of somatic mutation plus selection is known as affinity maturation. To a reader familiar with evolutionary biology, these two processes of clonal selection and affinity maturation are much akin to the (macro-)evolution of species. There

are a few basic differences however, between these immune processes and the evolution of species. Within the immune system, somatic cells reproduce in an asexual form (there is no crossover of genetic material during cell mitosis), the mutation suffered by an immune cell is proportional to its affinity with the selective pathogen (the higher the affinity, the smaller the mutation rate), and the number of progenies of each cell is also proportional to its affinity with the selective pathogen (the higher the affinity, the higher the number of progenies). Evolution in the immune system occurs within the organism and, thus it can be viewed as a micro-evolutionary process. As we know, in fact, immunology suggests that the natural Immune System (IS) has to assure recognition of each potentially dangerous molecule or substance, generically called antigen (Ag), by antibodies (Ab). The IS first recognizes an antigen as "dangerous" or external invaders and then adapts (by affinity maturation) its response to eliminate the threat. To detect an antigen, the IS activates a recognition process. In vertebrate organisms, this task is accomplished by the complex machinery made by cellular interactions and molecular productions. The main features of the clonal selection theory that will be explored in this chapter are [14]]:

- Proliferation and differentiation on stimulation of cells with antigens;
- Generation of new random genetic changes, subsequently expressed as diverse antibody patterns, by a form of accelerated somatic mutation (a process called affinity maturation);
- Elimination of newly differentiated lymphocytes carrying low affinity antigenic receptors.

To illustrate the adaptive immune learning mechanism, consider that an antigen *Ag1* is introduced at time zero and it finds a few specific antibodies within the animal (see Fig. 1.2). After a lag phase, the antibody against antigen *Ag1* appears and its concentration rises up to a certain level, and then starts to decline (*primary response*). When another antigen *Ag2* is introduced, no antibody is present, showing the specificity of the antibody response [14]. On the other hand,

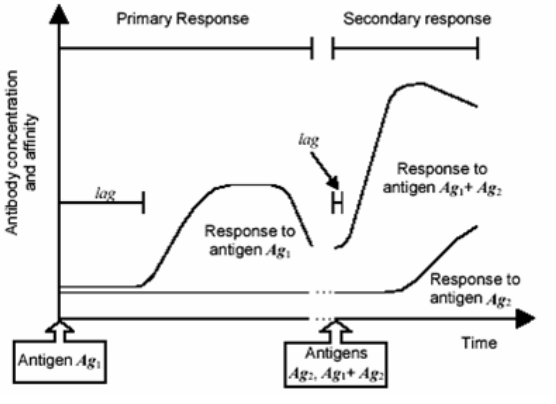


Fig. 1.2. Immune response plotted as antibody concentration over time

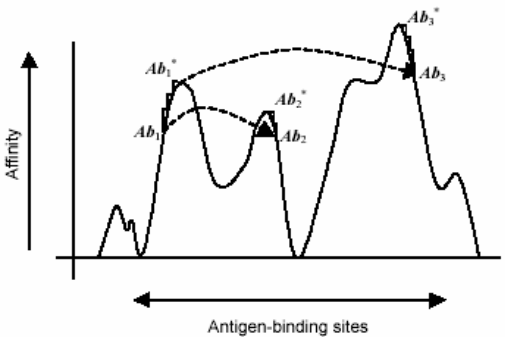


Fig. 1.3. Antibody affinity as function of the specific antigen binding site

one important characteristic of the immune memory is that it is associative: *B cells* adapted to a certain type of antigen *Ag1* presents a faster and more efficient secondary response not only to *Ag1*, but also to any structurally related antigen *Ag1 + Ag2*. This phenomenon is called immunological cross-reaction, or cross-reactive response. This associative memory is contained in the process of vaccination and is called *generalization capability*, or simply generalization, in other artificial intelligence fields, like neural networks [14].

Receptor editing offers the ability to escape from local optima on an affinity landscape. Fig. 1.3 illustrates this idea by considering all possible antigen-binding sites depicted the x-axis, with the most similar ones adjacent to each other. The Ag-Ab affinity is shown on the y-axis. If we consider a particular antibody (*Ab1*) selected during a primary response, then point mutations allow the immune system to explore local areas around Ab1 by making small steps towards an antibody with higher affinity, leading to a local optimum (*Ab1**). Because mutations with lower affinity are lost, the antibodies can not go down the hill. Receptor editing allows an antibody to take large steps through the landscape, landing in a locale where the affinity might be lower (*Ab2*). However, occasionally the leap will lead to an antibody on the side of a hill where the climbing region is more promising (*Ab3*), reaching the global optimum. From this locale, point mutations can drive the antibody to the top of the hill (*Ab3**). In conclusion, point mutations are good for exploring local regions, while editing may rescue immune responses stuck on unsatisfactory local optima.

Computational immunology is the research field that attempts to reproduce *in silico* the behavior of the natural IS. From this approach, the new field of Artificial Immune Systems (AIS) attempts to use theories, principles, and concepts of modern immunology to design immunity-based system applications in science and engineering [14]. AIS are adaptive systems in which learning takes place using evolutionary mechanisms similar to biological evolution. These different research areas are tied together: the more we learn from *in silico* modeling of natural systems, the better we are able to exploit ideas for computer science and engineering applications.

Thus one wants, first, to understand the dynamics of such complex behavior when they face antigenic attack, and second, one wishes to develop new algorithms that mimic the natural IS under study. Thus the final system may have a good ability to solve computational problems otherwise difficult to be solved by conventional specialized algorithms. The computational and predictive power of AIS offers researchers a promising approach for trying to solve well known and challenging problems like knowledge discovery from huge biological databases (e.g. coming from high throughput platforms) as well as protein folding or function prediction and multiple sequence alignment.

1.2 Type-2 Fuzzy Systems

Type-1 fuzzy sets are characterized by crisp grades of the membership function however, for some reasons, it could be very hard to find the exact membership function for a given fuzzy set and, as a consequence, it is hard to determine an exact membership level for each linguistic variable of the defined universe. It is then necessary to further fuzzify the knowledge base and this is possible only by using fuzzy sets that are fuzzy themselves [33]. Type-2 fuzzy sets are characterized by membership grades that are represented by values in the interval $[0, 1]$. At each value of the primary variable the membership is a function (and not just a point value), also called secondary membership function, whose domain the primary membership, is in the interval $[0, 1]$ and whose range secondary grades may also be in $[0, 1]$. We can assume, then, that the membership function of a Type-2 Fuzzy Set is three dimensional (see Fig. 1.4). This is a real plus to the theory of Type-1 Fuzzy Sets since it should be evident that such sets are useful in circumstances where uncertainty prevents us from obtaining a sufficiently clear knowledge on the process. As an example we consider the the well known case of *eye contact* [33]. Let us put the eye contact on a scale of values that goes from 0 to 10. One can say that a term of this universe can be '*some eye contact*'. Suppose

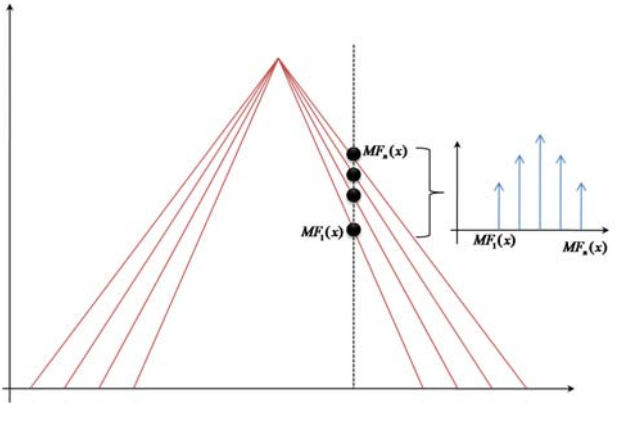


Fig. 1.4. Triangular MFs for a T2FS

we interviewed 100 men and women asking them to set boundaries for this measure on the $0 - -10$ scale. It is unlikely we will get the same results from all of them because words mean different things to different people and this situation is rather frequent even in specialized field like medicine. One approach to using the 100 sets of two end-points is to average the end-point data and to use the average values for the interval associated with some eye contact. We could then construct a triangular (other shapes could be used) membership function (MF), $MF(x)$, whose base end-points (on the x-axis) are at the two average values and whose apex is midway between the two end-points. This type-1 triangular MF can be displayed in two-dimensions. Unfortunately, it has completely ignored the **uncertainties** associated with the two end-points. A second approach is to make use of the average values and the standard deviations for the two end-points. By doing this we are blurring the location of the two endpoints along the x-axis. Now locate triangles so that their base end-points can be anywhere in the intervals along the x-axis associated with the blurred average endpoints. Doing this leads to a continuum of triangular MFs sitting on the x-axis, e.g. picture a whole bunch of triangles all having the same apex point but different base points, as in Fig. 1.4. For purposes of this discussion, suppose there are exactly 100 (N) such triangles. Then at each value of x, there can be up to N MF values, $MF_1(x), MF_2(x), \cdots, MF_N(x)$. Let us assign a weight to each of the possible MF values, say $w_{x1}, w_{x2}, \cdots, w_{xN}$ (see Fig. 1.4). We can think of these weights as the possibilities associated with each triangle at this value of x. At each x, the MF is itself a function -the secondary MF- ($MF_i(x)$, w_{xi}), where $i = 1, \cdots, N$. Consequently, the resulting type-2 MF is three-dimensional. For more details on T2 Fuzzy Sets the reader is referred to [32] and [28]. From the description we have provided it should be evident that uncertainty handling is a key point of these approaches. Uncertainty plays a major role in bio-medicine and biomedical science since most of the research carried out in this field is experimental and is affected by measurements associated errors. This is why we recently proposed a novel approach to data mining in bioinformatics that tries to face these problems using a coherent algorithmic model. In the next paragraphs we will describe type-1 and type-2 based fuzzy systems for rule inference from bioinformatic databases. We will provide a detailed description of both starting from the type-1.

1.3 Fuzzy-Immunity Based Data Mining Systems in Bioinformatics

Recent advances in active fields of research like biotechnology and electronics allowed biomedical research to make a significant step forward in the acquisition of fundamental tools for the elucidation of complex bio-processes like the ones behind cancer or Alzheimer disease. The advent of High-Throughput (HT) platforms has revolutionized the way researchers working in life sciences thought at their role in experiments. HT devices allowed researchers to concentrate on more important tasks like experimental design and results interpretation at the

same time allowing him to ignore the hundreds when not thousands of repeats of the same protocols for the different patients or mRNA sequences for instance. Microarrays are, probably, one of the most evident examples of this change of perspectives: gene expression evaluation for a panel of even only a few tens of genes took several days to be completed before their introduction, now we are able to obtain gene expression level for thousands of genes in the time of an overnight hybridization. Together with expression microarrays we can mention copy number monitoring microarrays (commonly referred to as aCGH technique), High-Throughput Sequencers, and Mass Spectrometers. In the next sections we will go through a brief analysis of the main open problems in bioinformatics and will discuss about how they can be addressed using immunity based data mining algorithms. A short introduction on data mining principles and potentialities is given in order to help unexperienced readers understanding concepts behind statements.

1.3.1 Data Bases and Information Retrieval in Biology

Devices coming from the integration of experiences gained in diverse fields like physics, chemistry, biology and engineering, in this way helped researchers in boosting their work and in quickly obtaining results of their experiments. The capabilities of these different kinds of approach pushed the interest for the establishment of data repositories for newly generated results. Data-bases entered the world of biology. Larger and larger amounts of data started to fill public databases (leaving apart literature databases which, of course, need a separated analysis) giving rise to what we can rename "Moore's law in biology" [46] (that just like the original Moore's law in electronics, models future progress in biotechnology [18]). However the main advantages provided by novel devices soon revealed to be their main weak point. The availability of large amount of data as results did not yield of information drawn from these data; this phenomenon characterized both early and more recent years in life sciences research bringing to the so-called "gap". Roughly speaking, researchers indicate, with this term, an estimate of the difference between the amount of available data and the amount of these data that have been sufficiently interpreted [24]. In the recent years we have observed a worrying widening in this gap: this means that we are making quite large investments with a ROI (return on investments) that still keeps low. In order to maximize the information yield of each experiment several alternative solutions have been proposed being probably data warehousing the most successful. Data warehouses are the natural evolution of data bases; described for the first time by William Immon [53]. They are integrated, subject-oriented, time-variant and non-volatile data collection processes implemented with the precise aim to build a unique decision support system. The distinction between data bases and data warehouses is clear: as advanced data bases, data warehouse provide data analysis functionalities that ease the process of knowledge extraction from highly dense data repository. In this context significant experiences like the GEO (Gene Expression for Omnibus [4]), SMD (Stanford Microarray Database [17]) and ArrayExpress [7] have been gained. It is evident that data

warehouse can greatly help researchers in reducing the gap by providing a valuable aid in filling the last real hole in experimental processes automation: results interpretation.

1.3.2 Mining the Data: Converting Data to Knowledge

Data mining, also known as Knowledge Discovery in Data-bases (KDD) , has been defined as *"The nontrivial extraction of implicit, previously unknown, and potentially useful information from data"* [20] (a more practical definition of data mining will be given in the following section); it uses machine learning, statistical and visualization techniques to discover and present knowledge in a form easily comprehensible to humans. Data mining grew at the border line among statistics, computer science and artificial intelligence and soon became a golden tool to solve problems ranging from Customer Relationship Management (CRM [31]) to Decision Making Support in medicine [47]. Data mining in bioinformatics, then, can be considered as a useful tool for modeling complex processes allowing researchers speeding the pace towards treatments for diseases like cancer: for instance several works have successfully tried to exploit the potentialities of rule induction systems in breast cancer associated survival [30, 5] and cancer evolution modeling [35]. It can be argued that data mining was born from several diverse disciplines, in the effort of overcoming intrinsic limitations of the single approaches. It is particularly evident if we compare the expressive power of typical statistical inference approaches and propositional or first order logic on the other hand. Huge efforts have been spent, in the recent past, in order to speed up one of the central tasks in current research in bioinformatics, that is, the transformation process that converts *data* in *knowledge* passing through *information* [43]. Data mining software, then, became more and more common: researchers soon realized the valuable aid algorithms could have given to their researchers and the amount of paper describing algorithms for information extraction grew faster and faster [15, 44, 55]. Comprehensive software tools for data mining purposes are currently largely used in bioinformatics and include both open-source and proprietary solutions. Among commercial packages we can list SPSS, SAS, Clementine and E-Miner. Open source tools are well represented by:

- Weka [54]
- Rapid Miner (formerly YALE) [40]
- Orange [39]

In particular Weka has gained a relevant success in the field of data mining due to its flexibility and versatility. Thanks to these characteristics Weka has been customized and redistributed in several different flavors (BioWeka [23] devoted to biological sequences mining and Weka4WS [48], the GRID-enable Weka implementation). Due to a simple but efficient modular organization Weka allowed third-party developers to add functionalities to the core package. It is the case of "Weka Classification Algorithms" project managed by Jason Brownlee who has implemented several bio-inspired [8, 9, 29] data mining algorithm in a

customized version of Weka Classification Algorithms[1]. One of the most interesting aspects of this implementation consists in the presence of a wide variety of Artificial Immune System based data mining algorithms. Both the *black* and *white box* flavors are represented in the set of proposed algorithms. The distinction between black and white box algorithms will be described in the following paragraph, however it can be argued that white box approaches provide the user with tools to easily interpret the way it reached a certain results, on the contrary to what happens with black box algorithms (think at how complex is the interpretation of neural network predictions and how simple is interpreting rules induced from a dataset). Among black box Immunity based algorithm we can mention:

Clonalg

The Clonal Selection Algorithm, originally called CSA in [12], and renamed to CLONALG in [13] is said to be inspired by the following elements of the clonal selection theory:

- Maintenance of a specific memory set
- Selection and cloning of most stimulated antibodies
- Death of non-stimulated antibodies
- Affinity maturation (mutation)
- Re-selection of clones proportional to affinity with antigen
- Generation and maintenance of diversity

The goal of the algorithm is to develop a memory pool of antibodies that represents a solution to an engineering problem. In this case, an antibody represents an element of a solution or a single solution to the problem, and an antigen represents an element or evaluation of the problem space.

CSCA

The Clonal Selection Classifier Algorithm is an evolution of the concept behind Clonalg since it tries to maximize classification accuracy and minimize misclassification accuracy still using clonal selection paradigms.

Immunos

The Immunos [10] algorithm has been mentioned a number of times in AIS literature [49, 25, 50]. It is claimed as being one of the first immune-inspired classification systems. Immunos tries to mimic in a very precise way the mechanisms underlying immune response to antigen attacks and this has led to a quite complex classification system still under discussion.

AIRS

The Artificial Immune Recognition System [52] algorithm was one of the first AIS technique designed specifically and applied to classification problems. After

[1] http://sourceforge.net/projects/wekaclassalgos

an initialization phase the algorithm cycles through each antigen (record in the dataset) in order to select best fitting memory cells through a powerful resource competition stage.

On the other hand white box AIS based paradigms can be found in:

- IFRAIS
- AIS based rule induction with boosting

These approaches will be discussed in greater depth in the next section.

1.3.3 Algorithmic Approaches to Data-Mining in Biology

As previously stated data mining is an interdisciplinary research field, involving areas such as machine learning, statistics, databases, expert systems and data visualization, whose main goal is to extract knowledge (or patterns) from real-world data sets [19, 54]. This section focuses on the classification (supervised learning) task of data mining. In essence, the goal of the classification task is to assign each example (data instance or record) to a class, out of a predefined set of classes, based on the values of attributes describing that example. In the context of bioinformatics an example could be, for instance, a protein; the classes could be protein functions; and the attributes describing the protein could be, say, physico-chemical properties of the amino acids composing the protein. It is important that the attributes describing an example are relevant for predicting its class. Hence, it would be a mistake to use a clearly irrelevant attribute, say the name of the patient, as an attribute to predict whether or not a patient will get a certain disease. In bioinformatics, ideally, the classification model should satisfy two requirements. First, it should have a high predictive accuracy, or generalization ability, correctly predicting the class of new examples unseen during the training of the system. Second, it should be comprehensible to users (biologists), so that it can be interpreted in the context of existing biological knowledge and potentially further validated through new biological experiments. Concerning the issue of comprehensibility of the classification model discovered from the data, it should be noted that some classification algorithms are designed to maximize only predictive accuracy, representing the classification model in a way that cannot be understood by the user - therefore ignoring the comprehensibility requirement. Typical examples of algorithms in this category are support vector machines [51] and neural networks [26]. In this case the classification model is a "black box", which does not give the user any insight about the data or explanations about the classification of new examples. In contrast, some classification algorithms use a representation which is comprehensible to the user, therefore returning "knowledge" to the user. In this section we focus on one popular kind of comprehensible representation, namely IF-THEN classification rules, and algorithms that use this kind of representation are called rule induction algorithms [21]. In rule induction algorithms the classification model is represented by a set of classification rules. These rules are of the form: "IF antecedent THEN consequent", where the antecedent represents a conjunction of conditions and the consequent represents the class predicted for all examples

(data instances, records) that satisfy the antecedent. Each condition in the antecedent typically specifies a value or a range of values for a given attribute of the data being mined - e.g., "gender = female", "age < 21".

The first AIS for rule induction in the classification task of data mining was proposed in [3], and named IFRAIS (Induction of Fuzzy Rules with an Artificial Immune System). IFRAIS as well as IFRAIS2 will be discussed in the next section. In this section we just highlight that this system discovers fuzzy classification rules. Fuzzy rules are in general more natural and more comprehensible to human beings than crisp rules, and the fuzzy rule representation also has the ability of coping well with the uncertainties frequently associated with data in biological databases [41]. Other algorithms ased on AIS for rule induction are discussed in detail in [1, 11].

Artificial Immune Systems in Bio-medical Data Mining: IFRAIS and IFRAIS2

As mentioned earlier, IFRAIS as well as its Type-2 FS counterpart are AIS that designed to discover fuzzy classification rules from data. From now on we will refer to IFRAIS as the main ideas behind it remained unchanged in IFRAIS 2 unless otherwise stated.

Recall that the rule antecedent is formed by a conjunction of conditions. Each attribute can be either continuous (real-valued, e.g. the molecular weight of a protein) or categorical (nominal, e.g. the name of a species), as usual in data mining. Categorical attributes are inherently crisp, but continuous attributes are fuzzified by using a set of three linguistic terms (low, medium, high). Hence, in the case of continuous attributes, IFRAIS discovers fuzzy rules having conditions such as: "molecular weight is large". IFRAIS discovers fuzzy classification rules by using the sequential covering approach for rule induction algorithms [54]. This is an iterative process which starts with an empty set of rules and the full training set (containing all training examples). At each iteration, IFRAIS is run to discover the best possible classification rule for the current training set, which is then added to the set of discovered rules. Then the examples correctly covered by the discovered rule (i.e. the examples satisfying the antecedent of that rule and having the class predicted by the rule) are removed from the training set, so that a smaller training set is available for the next iteration. This process is repeated until all (or a large part of the) training examples have been covered by the discovered rules. In order to discover classification rules, IFRAIS uses essentially clonal selection and hypermutation procedures. The basic ideas are as follows. Each antibody corresponds to a candidate fuzzy classification rule. During an IFRAIS run, the better the classification accuracy of an antibody, the more likely it is to be selected for cloning. In addition, once an antibody is cloned, the rate of mutation of a clone is inversely proportional to the classification accuracy of the antibody. Hence, the principles of clonal selection and hypermutation drive the evolution of the population of antibody towards better and better classification rules. In IFRAIS2, on the other hand, we are interested in evolving terms with MF that are fuzzy themselves so we handle them using a pre-defined number

Table 1.1. Results of IFRAIS and IFRAIS 2 on several data sets of varying complexity

Dataset	IFRAIS2	IFRAIS
CRX	74.82%	69.65%
Monk	91.08%	87.26%
Wine	85.12%	83.26%
Breast Cancer aCGH [38]	87.86%	78.65%
Breast Cancer Gene Expression [34]	87.45%	82.73%

of MF for each term and we evolve vectors of these features in place of single attributes (e.g. vectors of mean values or cut-values of MF in place of a single mean or cut-value). In [34, 38] IFRAIS was successfully employed to discover fuzzy classification rules for female breast cancer familiarity profiling. IFRAIS' results were validated using statistical driven approaches using Gene Ontology through GO Miner [55]. Competitive results obtained by IFRAIS and IFRAIS 2 (Tab. 1.1 show a comparative study of the results of both IFRAIS and IFRAIS 2 on benchmark, as well as, on real world data sets) seem to encourage new efforts in this field. A biological interpretation of the results carried out using Gene Ontology is currently under investigation.

1.3.4 Application of AIS Based Data Mining in Bioinformatics

As we previously stated several examples of application of Fuzzy-AIS based data mining systems in bioinformatics can be retrieved in literature. Fuzzy and Artificial Immune Systems-derived algorithms have been employed in familiarity profiling [34], prognosis prediction [35] and estrogen receptor modeling [36] in breast cancer. For a brief comparative overview of the performances of these kinds of systems in the context of aCGH data analysis the reader is referred to [37]. For the AIS counterpart we should note that previously de Castro and colleagues focused on the use of Hierarchical Artificial Immune Network paradigm for the problem of gene expression clustering [6, 27] and for rearrangement study of gene expression [16]. Research currently being carried out by Alves and colleagues is mainly focused on the application of a multi-label Fuzzy-AIS based data mining system to the problem of protein function prediction [2].

1.4 Conclusions and Open Questions

In this chapter we have analyzed some applications of Fuzzy-Artificial Immune System based algorithms in bioinformatics. Of course this is only a partial outlook on the world of Fuzzy-AIS based approaches: interested readers can check references in order to obtain more detailed information about specific aspects of the proposed topics. Furthermore, given their infancy, Fuzzy-AIS are currently undergoing very fast changes resulting in a very dynamical field of research where tens of novel and promising projects are proposed in the time of some months. These aspects forced the authors to select a set of significant experiences to be used as examples of how the algorithms described herein can be

successfully used in the field of bioinformatics. After these necessary statements some conclusions. In this chapter we have learned how novel bio-inspired computational intelligence paradigms can be used in very diverse field of research in bioinformatics. As previously stated Fuzzy-AIS are considered a novel paradigm but they have been already able to reach significant results in highly complex context like knowledge discovery in data bases and gene signature prediction. Even if fuzzy-immune-inspired algorithms have been successfully employed in several diverse problems, there are still some strategic fields of research in which solutions seem to be far from being reached, just to name few:

- Gene networks inference;
- Disease profiling and evolution modeling.
- Diagnostic and prognostic disease signature development

These are only some of the most active areas of Fuzzy-AIS based research in bioinformatics. From a theoretical point of view it should be noted that some areas like *hybrid systems* in this field have been exploited with a limited systematic approach in bioinformatics: these areas deserve a comprehensive analytic approach. Readers interested in these promising aspects of the Fuzzy-AIS research in bioinformatics can find useful information in [42, 45].

References

1. Alatas, B., Akin, E.: Mining fuzzy classification rules using an artificial immune system with boosting. In: Eder, J., Haav, H.-M., Kalja, A., Penjam, J. (eds.) ADBIS 2005. LNCS, vol. 3631, pp. 283–293. Springer, Heidelberg (2005)
2. Alves, R.T.: An artificial immune system to hierarchical multi-label classification for predicting protein function, Ph.D. Qualifying Exam 42, Federal University of Technology of Paran-UTFPR, Curitiba, Brazil (2007)
3. Alves, R.T., Delgado, M.R., Lopes, H.S., Freitas, A.A.: An artificial immune system for fuzzy-rule induction in data mining. 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. 1011–1020. Springer, Heidelberg (2004)
4. Barrett, T., Troup, D.B., Wilhite, S.E., Ledoux, P., Rudnev, D., Evangelista, C., Kim, I.F., Soboleva, A., Tomashevsky, M., Edgar, R.: NCBI GEO: mining tens of millions of expression profiles–database and tools update. Nucleic Acids Res. 35, D760–D765 (2007)
5. Bevilacqua, V., Chiarappa, P., Mastronardi, G., Menolascina, F., Paradiso, A., Tommasi, S.: Identification of tumour evolution patterns by means of inductive logic programming. Journal - Genomics Proteomics and Bioinformatics (in press)
6. Bezerra, G.B., Canado, G.M.A., Menossi, M., de Castro, L.N., Von Zuben, F.J.: Recent advances in gene expression data clustering: a case study with comparative results. Genet. Mol. Res. 4(3), 514–524 (2005)
7. Brazma, A., Parkinson, H., Sarkans, U., Shojatalab, M., Vilo, J., Abeygunawardena, N., Holloway, E., Kapushesky, M., Kemmeren, P., Lara, G.G., Oezcimen, A., Rocca-Serra, P., Sansone, S.A.: ArrayExpress–a public repository for microarray gene expression data at the EBI. Nucleic Acids Res. 31(1), 68–71 (2003)

8. Brownlee, J.: Artificial immune recognition system (AIRS) - A review and analysis, Tech. Report ID: 1-01, Centre for Intelligent Systems and Complex Processes, Faculty of Information and Communication Technologies, Swinburne University of Technology, Victoria, Australia (2005)
9. Brownlee, J.: Clonal selection theory and CLONALG - the clonal selection classification algorithm (CSCA), Tech. Report 2-01, Centre for Intelligent Systems and Complex Processes, Faculty of Information and Communication Technologies, Swinburne University of Technology, Victoria, Australia (2005)
10. Carter, J.H.: The immune system as a model for classification and pattern recognition. Journal of the American Informatics Association 7, 28–41 (2000)
11. Castro, P.A.D., Coelho, G.P., Caetano, M.F., Von Zuben, F.J.: Designing ensembles of fuzzy classification systems: An immune-inspired approach. In: Jacob, C., Pilat, M.L., Bentley, P.J., Timmis, J.I. (eds.) ICARIS 2005. LNCS, vol. 3627, pp. 469–482. Springer, Heidelberg (2005)
12. de Castro, L.N., Von Zuben, F.J.: The clonal selection algorithm with engineering applications. In: Genetic and Evolutionary Computation Conference, Workshop on Artificial Immune Systems and Their Applications, pp. 36–37 (2000)
13. de Castro, L.N., Von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation 6, 239–251 (2002)
14. de Castro, L.N., Timmis, J.: Artificial immune systems: A new computational approach. Springer, Heidelberg (2002)
15. de la Nava, J.G., Santaella, D.F., Alba, J.C., Carazo, J.M., Trelles, O., Pascual-Montano, A.: Engene: The processing and exploratory analysis of gene expression data. Bioinformatics, 657–658 (2003)
16. de Sousa, J.S., de Gomes, L.C.T., Bezerra, G.B., de Castro, L.N., Von Zuben, F.J.: An immune-evolutionary algorithm for multiple rearrangements of gene expression data. Genetic Programming and Evolvable Machines 5(2), 157–179 (2004)
17. Demeter, J., Beauheim, C., Gollub, J., Hernandez-Boussard, T., Jin, H., Maier, D., Matese, J.C., Nitzberg, M., Wymore, F., Zachariah, Z.K., Brown, P.O., Sherlock, G., Ball, C.A.: The stanford microarray database: implementation of new analysis tools and open source release of software. Nucleic Acids Res., D766–D770 (2007)
18. Life 2.0. the new science of synthetic biology is poised between hype and hope. but its time will soon come, Economist (September 2006)
19. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R.: Advances in knowledge discovery and data mining. AAAI/MIT, Cambridge (1995)
20. Frawley, W., Piatetsky-Shapiro, G., Matheus, C.: Knowledge discovery in databases: An overview. AI Magazine 13(3), 57–70 (1992)
21. Freitas, A.A.: Data mining and knowledge discovery with evolutionary algorithms. Springer-Verlag, Berlin (2002)
22. Galea, M., Shen, Q.: Iterative vs simultaneous fuzzy rule induction. In: IEEE Conference on Fuzzy Systems, pp. 767–772 (2005)
23. Gewehr, J.E., Szugat, M., Zimmer, R.: BioWeka-extending the Weka framework for bioinformatics. bioinformatics 23(5), 651–653 (2007)
24. Grossman, R., Kamath, C., Kumar, V.: Data mining for scientific and engineering applications. Springer, Heidelberg (2001)
25. Hart, E.: Immunology as a metaphor for computational information processing: Fact or fiction?, Ph.D. thesis, University of Edinburgh (2002)
26. Haykin, S.: Neural networks - a comprehensive foundation, 2nd edn. Prentice Hall, Upper Saddle River (1999)

27. Hruschka, E.R., Campello, R.J.G.B., de Castro, L.N.: Evolving clusters in gene-expression data. Inf. Sci. 176(13), 1898–1927 (2006)
28. http://www.type2fuzzylogic.org/publications/
29. Brownlee, J.: Immunos-81 - the misunderstood artificial immune system, Tech. Report 3-01, Centre for Intelligent Systems and Complex Processes, Faculty of Information and Communication Technologies, Swinburne University of Technology, Victoria, Australia (2005)
30. Larranaga, P., Gallego, M.J., Sierra, B., Urkola, L., Michelena, M.J.: Bayesian networks, rule induction and logistic regression in the prediction of the survival of women suffering from breast cancer. In: Spanish Artificial Intelligence Conference, pp. 303–308 (1997)
31. Ledingham, D., Rigby, D.K.: CRM done right. Harvard Business Review (2004)
32. Mendel, J.M.: Advances in type-2 fuzzy sets and systems. Information Sciences 177(1), 84–110 (2007)
33. Mendel, J.M., John, R.I.: Type-2 fuzzy sets made simple. IEEE Trans. of Fuzzy Systems 10(2), 117–127 (2002)
34. Menolascina, F., Alves, R.T., Tommasi, S., Chiarappa, P., Delgado, M., Bevilacqua, V., Mastronardi, G., Freitas, A.A., Paradiso, A.: Fuzzy rule induction and artificial immune systems in female breast cancer familiarity profiling. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part III. LNCS (LNAI), vol. 4694, pp. 830–837. Springer, Heidelberg (2007)
35. Menolascina, F., Alves, R.T., Tommasi, S., Chiarappa, P., Delgado, M., Bevilacqua, V., Mastronardi, G., Freitas, A.A., Paradiso, A.: Improving female breast cancer prognosis by means of fuzzy rule induction with artificial immune systems. In: Proceedings of 2007 International Conference on Life System Modeling and Simulation, Shanghai, China, pp. 1–5 (2007)
36. Menolascina, F., Alves, R.T., et al.: Induction of fuzzy rules with artificial immune systems in aCGH based ER status breast cancer characterization. In: Genetic and Evolutionary Computation Conference (2007)
37. Menolascina, F., Tommasi, S., Chiarappa, P., Bevilacqua, V., Mastronardi, G., Paradiso, A.: Data mining techniques in acgh-based breast cancer subtype profiling: an immune perspective with comparative study. BMC Systems Biology (suppl. 1), 70 (2007)
38. Menolascina, F., Tommasi, S., Paradiso, A., Cortellino, M., Bevilacqua, V., Mastronardi, G.: Novel data mining techniques in aCGH based breast cancer subtypes proling: the biological perspective. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, Honolulu. US, pp. 9–16 (2007)
39. Michalski, R.S., Bratko, I., Kubat, M.: Machine learning and data mining: Methods and applications. Wiley, Chichester (1998)
40. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: YALE: Rapid prototyping for complex data mining tasks. In: 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 935–940. ACM Press, New York (2006)
41. Pedrycz, W., Gomide, F.: An introduction to fuzzy sets: Analysis and design. MIT Press, Cambridge (1998)
42. Polat, K., Sahan, S., Gunes, S.: A novel hybrid method based on artificial immune recognition system (airs) with fuzzy weighted pre-processing for thyroid disease diagnosis. Expert Systems with Applications: An International Journal 32(4), 1141–1147 (2007)

43. Pool, R., Esnayra, J.: Bioinformatics: Converting data to knowledge. Natl. Acad. Press, Washington (2003)
44. Reich, M., Liefeld, T., Gould, J., Lerner, J., Tamayo, P., Mesirov, J.P.: GenePattern 2.0. Nature Genetics 38(5), 500–501 (2006)
45. Sahan, S., Polat, K., Kodaz, H., Gunes, S.: A new hybrid method based on fuzzy-artificial immune system and k-nn algorithm for breast cancer diagnosis. Computers in Biology and Medicine 37(3), 415–423 (2007)
46. Scott, R.: Keynote Speech, TNTYN, San Francisco (2000)
47. Siadaty, M.S., Knaus, W.A.: Locating previously unknown patterns in data-mining results: a dual data- and knowledge-mining method. BMC Medical Informatics and Decision Making 6, 13 (2006)
48. Talia, D., Trunfio, P., Verta, O.: Weka4WS: A WSRF-enabled weka toolkit for distributed data mining on grids. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS, vol. 3721, pp. 309–320. Springer, Heidelberg (2005)
49. Timmis, J., Knight, T., De Castro, L.N., Hart, E.: An overview of artificial immune systems. In: Paton, R., Bolouri, H., Holcombe, M., Parish, J.H., Tateson, R. (eds.) Computation in Cells and Tissues: Perspectives and Tools for Thought, pp. 51–86. Springer, Heidelberg (2004)
50. Twycross, J.: An immune system approach to document classification, Master's thesis, University of Sussex (2002)
51. Vapnik, V.N.: The nature of statistical learning theory. Springer, Heidelberg (1995)
52. Watkins, A.B., Boggess, L.C.: A resource limited artificial immune classifier. In: Congress on Evolutionary Computation, vol. 1, p. 926–931 (2002)
53. Immon, W.H.: Building the data warehouse. John Wiley and Sons, New York (1996)
54. Witten, I.H., Frank, E.: Data mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Mateo (2005)
55. Zeeberg, B.R., Feng, W., Wang, G., Wang, M.D., Fojo, A.T., Sunshine, M., Narasimhan, S., Kane, D.W., Reinhold, W.C., Lababidi, S., Bussey, K.J., Riss, J., Barrett, J.C., Weinstein, J.N.: GoMiner: A resource for biological interpretation of genomic and proteomic data. Genome Biology 4(4), R28 (2003)

2

Fuzzy Genome Sequence Assembly for Single and Environmental Genomes

Sara Nasser, Adrienne Breland, Frederick C. Harris Jr., Monica Nicolescu, and Gregory L. Vert

Department of Computer Science & Engineering, University of Nevada Reno, Reno NV 89557, USA
{sara,brelanda,Fred.Harris,monica,gvert}@cse.unr.edu

Summary. Traditional methods obtain a microorganism's DNA by culturing it individually. Recent advances in genomics have lead to the procurement of DNA of more than one organism from its natural habitat. Indeed, natural microbial communities are often very complex with tens and hundreds of species. Assembling these genomes is a crucial step irrespective of the method of obtaining the DNA. This chapter presents fuzzy methods for multiple genome sequence assembly of cultured genomes (single organism) and environmental genomes (multiple organisms).

An optimal alignment of DNA genome fragments is based on several factors, such as the quality of bases and the length of overlap. Factors such as quality indicate if the data is high quality or an experimental error. We propose a sequence assembly solution based on fuzzy logic, which allows for tolerance of inexactness or errors in fragment matching and that can be used for improved assembly.

We propose fuzzy classification using modified fuzzy weighted averages to classify fragments belonging to different organisms within an environmental genome population. Our proposed approach uses DNA-based signatures such as GC content and nucleotide frequencies as features for the classification. This divide-and-conquer strategy also improves performance on larger datasets. We evaluate our method on artificially created environmental genomes to test various combinations of organisms and on an environmental genome.

2.1 Introduction

DNA is the building block of all life on this planet, from single cell microscopic bacteria to more advanced creatures such as humans. Twenty years after the DNA code was cracked, Frederic Sanger, a Nobel Laureate, invented the chain termination method of DNA sequencing, also known as the Dideoxy termination method or the Sanger method [39]. His research paved the way to a technique to obtain DNA sequences and to the first genome sequence assembly, Bacteriophage ϕX174 [38]. In 1990 the Human Genome Project was announced, which sought to sequence the billions of nucleotides present in human DNA and was completed in 2003, two years before its projected date. In 1993 The Institute for Genome Research (TIGR) decided to use the TIGR EST algorithm which is based on whole-genome shotgun sequencing method to assemble a microbial

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 19–44.
springerlink.com

genome. Thus in 1995 at TIGR, *Haemophilus influenzae* was sequenced, becoming the first genome to be sequenced entirely [12]. Shotgun sequencing was first demonstrated to close a genome in this paper. The assembly of *H. influenzae* proved the potential of shotgun sequencing and thus lead to subsequent projects that would be based on shotgun sequencing. There were two major advancements in technology that lead the to complete sequencing of the Human Genome and the *H. influenzae*: shotgun sequencing and the use of computational techniques. This brief history gives insight into the advancements that were made in sequencing using computational techniques. For more history on the evolution of DNA sequencing and timelines of genome sequencing projects, refer to [7] and [10].

A DNA strand consists of four nucleotides: Adenine(A), Cytosine(C), Guanine(G) and Thymine(T). Genome sequencing is figuring out the order of DNA nucleotides, or bases, in a genome that make up an organism's DNA. Genome sequences are large in size and can range from several million base pairs in prokaryotes to billions of base pairs in eukaryotes. For example, Wolbachia genome, a bacteria has 126 million base pairs (Mb), Arabidopsis thaliana, a plant has 120Mb, and the human genome is 3.2 billion base pairs. The whole genome cannot be sequenced all at once because available methods of DNA sequencing can handle only short stretches of DNA at a time. Although genomes vary in size from millions of nucleotides in bacteria to billions of nucleotides in humans, the chemical reactions researchers use to decode the DNA base pairs are accurate for shorted lengths [32]. Genomes are cut at random positions then cloned to obtain the smaller fragments, also known as shotgun sequences. Obtaining shotgun sequences has allowed sequencing projects to proceed at a much faster rate, thus expanding the scope of the realistic sequencing venture [10]. Sequencing DNA using the shotgun method led to the completion of several organism genomes, including human, mouse, fruit fly and several microbes, such as *Wolbachia* genome and *H. influenzae*.

Microorganisms live in communities, and their structure and behavior is influenced by their habitat. Most microorganisms genomes are known from pure cultures of organisms isolated from the environment, be it a natural organism-associated (i.e, human) or artificial system. Cultivation-based approaches miss majority of the diversity that exists however, such that development of cultivation-free methods has been implied. In the past, microbial DNA was sequenced by culturing microorganisms in a controlled environment. Cultivating these organisms did not reveal enough information about these communities of organisms. Invitro cultivation methods allow the extraction of DNA from only a limited selection of microbial species that can grow in artificial environments. These methods do little to characterize the properties of globally distributed microbes, because the vast majority of them have not been cultured.

New techniques in genomic sciences have emerged that allow an organism to be studied in its natural habitat as part of a community. Research has broadened from studying single species to understanding microbial systems and their adaptations to natural environments. These techniques have been achieved by

developing methods that can sequence mixed DNA directly from environmental samples [2, 36].

Whole-genome shotgun sequencing of environmental DNA gained attention as a powerful method for revealing genomic sequences from various organisms in natural environments [2, 41]. An organism's DNA was not only sliced into small fragments but also mixed with other organisms' DNA fragments, thus creating a huge population of fragments, initial efforts were with long fragments ranging from 40Kb - 150Kb in fosmid or BAC libraries. Even though DNA fragments from diverse populations can be gathered together at the same time, they need to be assembled in order for us to make meaningful conclusions.

There are several approaches that are designed for examining a single organism, but there is a need for tools that are specific for community-level analysis. In this chapter, we propose methods of sequence classification and assembly for a metagenomic population. Sequence classification is a process of grouping genome fragments into classes based on their similarities. The proposed method aims to use an approximate method based on fuzzy logic to classify genome fragments into groups and then perform assembly.

The rest of the chapter is structured as follows: Sect. 2.2 presents background information on computational biology and DNA sequence assembly, including a survey of the related literature. Sect. 2.3 presents the fuzzy solution for sequence assembly. Sect. 2.4 presents taxonomical classification methods for metagenome fragments. Improvements achieved due to signatures and a new technique of fuzzy classification, in addition to results attained, are included in Sect. 2.3 and Sect. 2.4. Conclusions and a look into future directions are presented in Sect. 2.5.

2.2 Background

2.2.1 Genome Sequence Assembly

Several concepts and terms from genomic sciences that are used this in chapter are informally defined below. There are several books on computational biology that provide detailed explanations of the terms listed below [1, 50].

Definition 1. *Base Pair: Two nucleotides on a paired double-helix-structured DNA strand. These two nucleotides are complements of each other.*

Definition 2. *DNA Fragment/Read: A section of the genome sequence of nucleotides that forms a DNA strand.*

Definition 3. *Contig (Contiguous Sequence): A consensus sequence reated by overlapping two or more sub-sequences or fragments.*

Definition 4. *Nucleotide Frequencies: The measure of occurrences of nucleotide pairs of a specified length.*

Definition 5. *Metagenome: Genome sequences containing an unspecified number of microbial organisms directly obtained from the natural habitat.*

The problem of sequence assembly is acquiring data and assembling the DNA fragments or sequences into an entire genome sequence. Available chemical technologies for sequences produce short fragments of DNA sequences (40 Kbp -1000 Kbp) depending on the technology. Sequencing machines cannot read entire DNA, and can only work on small stretches at a time. There are two important aspects to understanding the problems that arise in genome assembly: the genome is cut into smaller portions, and fragments or sequences are cut at random positions. To obtain the original sequence these fragments need to be combined by determining overlaps between fragments. Thus, portions of the fragments need to appear more than once. Multiple copies of original sequences are made to ensure the entire sequence is covered. This process is generally referred to as coverage of nX, where n is the number of copies and X is the sequence. Coverage of 8X or 10X is widely accepted and it has been shown it is sufficient to reconstruct the entire sequence. Thus for a genome sequence of length 4(million)MB, if the sequence fragments of length around 500 bp are generated we need 80,000 sequences.

Following the sequencing process, an assembler pieces together the many overlapping bases and reconstructs the original sequence [32]. The process explained above is known as the whole-genome shotgun method. There are three main steps involved in the assembly of sequences. The first step, Sequencing, breaks the genomic DNA into fragments by sonication, a technique which uses high-frequency sound waves to make random cuts in DNA molecules [4]. In the assembly phase the sequences are combined to form contiguous sequences. The final phase is finishing, in this phase contigs are joined by closing physical gaps. Closing is a time consuming process, which can be improved by using more than one clone libraries. Clone libraries are prepared using different vectors. As different vectors clone sequences differently, using more than one vector can help improve coverage. Fragments that could not be cloned by one vector could be cloned by the other. Thus gaps could be reduced as overall coverage increases when sequences are generated using different vectors.

Sequencing of an organism's DNA is a labor-intensive task, made possible by recent advances in automated DNA sequencing technology.Even though automated DNA sequencing technology made it possible to sequence genomes, several other problems exist. The sequence read from a machine is not always 100% correct; it may contain experimental errors. The process of acquisition of genome sequence data may lead to the insertion of certain discrepancies in the sequences, known as base-calling errors. The actual DNA sequence is read as a frequency signal, which is converted to represent the character sequence representing the four nucleotides as A, C, G and T. PHRED is a popular tool for reading signals and assigning quality scores [11]. In this scoring technique each nucleotide is given a score based on the strength of the base, a high score implying a higher probability of the base being correct. Low scores are assigned to bases that have less probabilty of being true. Sequences at the ends tend to have weak signals that make it difficult to identify them. Thus the base is assigned to the closest match and marked as a low quality base.

Base-calling errors create additional problems during assembly. These differences are categorized into three groups: insertions, deletions (indels) and replacements. Another well known problem with the sequences is that they contain repetitive sections also known as repeats. All the parameters mentioned above make assembly an approximation problem.

The most popular approach to DNA fragment assembly has been to iteratively find the best overlap between all fragment pairs until an acceptable final layout is determined. If enough fragments are sequenced and their sampling is sufficiently random across the source, the process should determine the source by finding sequence overlaps among the bases of fragments that were sampled from overlapping stretches [14]. In current genome sequencing tasks, the number of fragments is usually numerous, and the degree of computation required increases exponentially. Being essentially an NP-hard problem, many different approaches with varied parameters and matching schemes have been explored to save computation time.

The earliest approach to find solutions using the shotgun sequence approach was to find the shortest common superstring from a set of sequences. Current approaches use pairwise sequence alignment as a method and instead of obtaining the shortest superstring, the longest common substring is used. To obtain the common substrings of two sequences, we are required to consider all possible substrings of the given sequences. The substring with the longest overlap is known as the longest common sequence (LCS). Finding the LCS for all possible sequences is an NP-hard problem. Thus, a brute-force approach is not feasible. Dynamic programming solves problems by combining the solutions to subproblems to reduce the runtime of algorithms containing overlapping subproblems and optimal substructures [9]. Using dynamic programming, we can find a polynomial-time solution for the LCS problem. Therefore, dynamic-programming-based approaches are the most routinely used approaches in sequence assembly and alignment.

Other techniques for finding the LCS include suffix trees and greedy approaches. A suffix tree is a data structure that uses suffix information for fast processing of string problems. A suffix tree can be constructed in linear time using the Ukkonen algorithm [47]. Even though suffix trees are a linear answer to sequence comparison problems, they are not good at storing and handling large datasets. Greedy algorithms are shown to be much faster than traditional dynamic programming in the presence of sequencing errors [54]. Greedy paradigms, applied in popular assemblers such as TIGR [42], Phrap [15], and CAP3 [17], are relatively easy to implement, but they are inherently local in nature and ignore long-range relationships between reads that could be useful in detecting and resolving repeats [32]. Greedy and hill-climbing approaches generally find a local optimal and thus the global solution could be missed. Additionally, these algorithms work with specific kinds of errors and cannot be generalized; they also become difficult to implement on larger datasets.

Unlike greedy approaches the overlap–layout–consensus mechanism considers all possible solutions before selecting the consensus overlap. An application of

graph theory is found in [20] in which fragment reads are represented by nodes and an overlap between two fragments is represented by an edge. Paths are constructed through the graph such that each path forms a contig. The paths are then cleaned by resolving and removing any problems such as intersecting paths, and consensus sequences are constructed following the paths. One of the major problems of this approach to DNA sequence assembly is the extensive computation requirement. Fragment assembly performed with the overlap–layout–consensus approach becomes inefficient with an increasing number of fragments.

Even with the algorithmic improvements, additional reductions to the search space in fragment assembly problems are routinely employed. Pre-assembly clustering of fragments may be viewed as a more structured form of fragment thinning before alignment comparisons are made. Clustering is a process of grouping objects into like groups based on some measure of similarity. Clustering or classification can be achieved by several techniques such as K-means and artificial neural networks. A divide-and-conquer strategy for sequence assembly based on average mutual information is described in [30].

2.2.2 Environmental Genomics

Molecular biology has impacted microbiology by shifting the focus away from clonal isolates and toward the estimated 99% of microbial species that cannot currently be cultivated [6, 18, 34]. As an illustration, traditional culture and PCR-based techniques showed a bias of *Firmicutes* and *Bacteroides* as the most abundant microbial groups in the human gastrointestinal (GI) tract. Metagenomic sampling has revealed that *Actinobacteria* and *Archaea* are actually most prolific [23].

Metagenomic data can be ecosystem or organism associated: ecosystem associated metagenome contains DNA of microbes obtained from an environmental sample and an organism associated metagenome contains DNA from organisms. For example, the Sargasso sea project, an ecosystem associated metagenomes containfiltering of sea water. These samples contained large amounts of novel genetic information, including 148 new bacterial phylotypes, 1.2 million new genes, and 782 new rhodopsin-like photoreceptors [48]. A similar metagenomic project giving new insight into naturally existing bacterial systems was the sampling biofilm from of an underground acid mine drainage [46]. Because this sample was from a system with low complexity, almost all DNA from present species were completely reconstructed, allowing the examination of strain differences and naturally forming lineages. It also enabled access to the full gene complement for at least two species, providing detailed information such as metabolic pathways and heavy metal resistance. Soil samples and the mouse GI tract are some other published metagenomic projects [36, 45].

Closely related organisms can contain remarkable genomic diversity, as was shown for some bacteria [49]. These variations, even though few, can result in different metabolic characteristics. Extracting these variations is one of the key ideas to further processing of the metagenomes. The genomic diversity between metagenome samples is extracted and used as a marker to separate data phylogenetically.

2.2.3 Phylogenetic Classification Using Signatures

The metagenomic approach of acquiring DNA fragments often lacks suitable phylogenetic marker genes, rendering the identification of clones that are likely to originate from the same genome difficult or impossible [44]. Separating the fragments in a metagenomic sample and reconstructing them is a complex process. Identification of certain features can distinguish one genome from another in some circumstances. Organisms within a metagenome population can belong to different ranks in the taxonomy, for example they could belong to different domains or could be from the same species. For example, the acidmine drainage data consists samples that belong to archeal and bacterial domains. A metagenome population can contain a large number of organisms that could either be very diverse, that is not closely related or it could be constrained to strains or species that are closely related. Thus complexity of a metagenome can also dictate the classification accuracy. Metagenome complexity can be measured with three different parameters: taxonomic relation, evenness, and richness. Visualization these is complex for example there are several ranks within taxonomy. This subsection describes the DNA signatures that can be employed in identification of differences within a metagenome.

DNA signatures are specific patterns that are observed within a DNA strand. These patterns can be observed in specific regions such as coding regions or can be observed throughout a genome. There have been several studies on the patterns found in DNA sequences. Biological sequences contain patterns that can lead to discoveries about the sequences. Two kinds of signatures are important to our discussion: GC content and oligonucleotide frequencies. Oligonucleotide composition within a genome contains bias, making certain patterns appear several times within the genome. These oligonucleotide usage patterns are known to be species-specific [19].

The four nucleotides of a DNA strand (A, C, G, and T) have hydrogen bonds between them. The nucleotide A bonds specifically with T and the nucleotide C bonds with G. AT pairs have two hydrogen bonds and GC pairs have three hydrogen bonds, making the GC bond thermostable. Thus, the GC content in an organism can sometimes be used to determine certain characteristics about that organism. Organisms are generally biased in the distribution of A, C, G and T. Certain organisms contain higher percentages of GC and are thus known as GC rich, while some other organisms are dominated by AT and are known as AT rich. This fundamental property of organisms can be used in separating one organism from another.

Another signature that has been used frequently for analysis of genome sequences is the oligonucleotide frequencies, which is a measure of the occurrence of words of fixed sizes in the genomic sequence. Oligonucleotides are short sequences of nucleotides generally of length less than 20. Nucleotide frequencies have been extensively used for grouping species or for differentiation of species. Specific details about obtaining nucleotide frequencies will be covered in Sect. 2.4.

2.2.4 Fuzzy Logic

The concept of fuzzy logic and approximate reasoning was introduced in 1975 [53]. Fuzzy logic formalizes an intuitive theory based on human approximation, which is by definition imprecise or vague. Fuzzy set theory allows classification of entities into categories by establishing degrees of weak or strong membership. A fuzzy set F is given by

$$
\begin{aligned}
&F = \{\mu_F(x) \mid x \in X, \mu_F(x) \in [0,1]\} \\
&where: \\
&x = \ a\ given\ element \\
&\mu_F(x) \ = \ fuzzy\ set\ membership\ function \\
&X = \ the\ Universe\ of\ Discourse
\end{aligned}
$$

The fuzzy set membership function, $\mu_F(x)$, returns a membership value between 0 and 1(inclusive) that signifies the degree of membership.

Fuzzy logic has been used in several engineering applications. Fuzzy approaches to bioinformatics have been explored to some extent. Even though the application of fuzzy logic is not widely used, it has begun to gain popularity. An application to ontology similarity using fuzzy logic was presented in [52]. Fuzzy logic also been applied to classification problems in computational biology. A modified fuzzy K-means clustering was used to identify overlapping clusters of yeast genes [13]. A model for creating fuzzy set theory for nucleotides was proposed by Sadegh-Zadeh [37]. In this model a fuzzy polynucleotide space is made to measure the degree of difference and similarity between sequences of nucleic acids. Alignment of sequences has different specifications, and thus, alignment tools are not suitable for assembly purposes. Assembly of sequences is influenced by several factors besides the sequence chain. Therefore, there is a need for an approximation method that takes into consideration all the factors for assembling sequences. Specific fuzzy applications for sequence assembly and classification will be covered in Sects. 2.3 and 2.4.

2.3 Fuzzy Genome Sequence Assembly

DNA sequence assembly can be viewed as the process of finishing a puzzle, where the pieces of the puzzle are DNA subsequences. Although a puzzle has pieces that fit together well, the pieces of a DNA puzzle do not fit together precisely; the ends can be ragged and some pieces are missing, thus making it difficult, and sometimes nearly impossible, to complete the puzzle. Hence, we need methods or rules to determine optimally which piece fits with another piece. The problem of sequence assembly is one of obtaining approximate matches through onsensus. A consensus sequence is constructed through approximate matches by following an overlap and consensus scheme [31] as illustrated in Fig. 2.1.

In current genome sequencing tasks, the number of fragments is usually large, and the degree of computation required increases exponentially. Being essentially an NP-hard problem, many different approaches with varied parameters

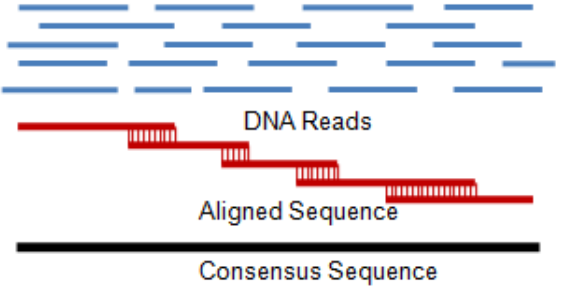


Fig. 2.1. Whole Genome Sequencing Process of Creating Contigs from Fragment Reads

and matching schemes have been explored which can, among other things, save computation time. Finding the longest common subsequence (LCS) between fragments is the key to the process of sequence assembly.

In this section, an approximate matching scheme based on fuzzy membership functions is presented. Several parameters are considered to create an optimal assembly. Then a divide-and-conquer strategy is presented to speed up the assembly by dividing the sequences into classes. Assembling sequences is accomplished by first grouping the sequences into clusters so that sequences in a cluster have high similarity with one another and sequences between two clusters are less similar.

2.3.1 Previous Techniques

Dynamic programming has been extensively used to determine the LCS as it reduces the NP-hard problem to a time complexity of $\Theta(n^2)$. The method is simple and useful in finding the LCS that may have mismatches or gaps. The Smith-Waterman algorithm, an application of dynamic programming to find the LCS for multiple sequence alignment, is one of the most prominent algorithms used in sequence assembly programs. The algorithm gained popularity because it reduces the number of searches required; more details of the algorithm can be found in [40]. Most of the earlier assemblers have crisp bounds and do not adapt to the datasets. For example, a dataset can contain all, or a significant number of, low quality reads. Some of the assemblers clip low quality regions, which will result in, most of the regions getting clipped and thus, not used in assembly. However, if the assembler can adapt itself and allow a new threshold for low quality, this problem can be avoided. Due to its applicability to problems that do not require hard solutions, fuzzy logic has been widely used in various fields to provide flexibility to classical algorithms. Thus, approximate sequence assembly is a good candidate for fuzzy logic. In the next subsection a non-greedy approach is presented, based on approximate sequence matching using fuzzy logic.

2.3.2 Sequence Assembly

The sequence assembly problem is tackled using two different approaches: the first module performs fuzzy sequence assembly, and the second module performs

a fuzzy divide-and-conquer strategy for assembly. The divide-and-conquer strategy uses a fuzzy membership function to divide genome sequences into groups, reducing the number of comparisons and performing meaningful assembly. The fuzzy functions used in this subsection are a modified version of the fuzzy genome sequencing assembler described in [25].

Longest Common Subsequence with Fuzzy Logic

Sequence assembly requires creating contigs from fragment reads. The longest subsequence with fewest insertions or deletions (indels) is ideal. Since an exhaustive search is not applicable for this problem, a time constraint is also placed on the solution. One of the problems with existing techniques for sequence assembly is that they have crisp bounds. The user has to specify the parameters for the program, such as minimum score and minimum match. lmost all existing techniques provide user-defined thresholds; the user generally runs the program several times to obtain optimal results. In such cases it is better to determine empirically the ideal cut-off point or the threshold. For example, assume that a cutoff value for the maximum gap allowed is 30 bases and that there are fairly large numbers of sequences with gaps of 31 and 32. Due to the fact that these techniques allow for crisp matches only, these potentially important sequences would be excluded. Alternatively, we can represent a gap of 30 and lower with a fuzzy confidence value of 1, which is for crisp results. Sequences with gaps that are very close to 30, like 31, can have a fuzzy value of 0.98. In this case, the user does not have to preprocess the data, change parameters and run the program several times.

The approach starts by acquiring the LCS of given sequence fragments using dynamic programming (details of the fuzzy LCS technique can be found in [25]). The optimal subsequence can be a perfect match, or the user may choose to tolerate indels. These criteria can depend on the user, the source of the data or the quality of the data. There are several factors that determine if two subsequences have an optimal overlap. We propose a method in which we select multiple subsequences and then, based on fuzzy parameters, select the optimal solution. The novelty of our method is that it uses more parameters of the sequence besides the length of overlap, and we believe that these parameters can lead to a better sequence. The sequence satisfying the aggregate overall requirement is selected. The process starts with LCS and selecting all the subsequences that satisfy the minimum length required as given in (2.1). The threshold is a function of the length of the LCS.

$$length \geq threshold, \quad threshold = f(length(LCS)) \tag{2.1}$$

In (2.1) *length* is the size of overlap, and *threshold* determines the minimum length required. The selection of the optimal subsequence is done using fuzzy similarity measures in constant time; therefore, the complexity of the algorithm is same as the complexity of dynamic programming, which is $\Theta(mn)$ for any two subsequences of length m and n. After the LCS is obtained we need to determine the other factors that influence assembly. The following subsection lists the descriptions along with the characteristic functions for each of the parameters.

Fuzzy Similarity Measures

Fuzzy similarity measures and the concept created by this research are an important step in creating a contig from two subsequences or finding an overlap between two sequences. The following subsections describe the fuzzy functions utilized in our approach for assembly.

Length of Overlap

The first similarity measure is the length of the match or length of overlap μ_{lo}, which includes indels and replacements. A higher overlap is better because it generates a longer contig; thus, this function aims at maximizing overlap. The membership function for this measure is defined as:

$$\mu_{lo}(s1, s2) = \begin{cases} 1, if |overlap(s1, s2)| = max|overlap(s1, s2)| \\ 0, if |overlap(s1, s2)| = 0 \\ |overlap(s1, s2)|/max|overlap(s1, s2)|, otherwise \end{cases}$$

Here, $|overlap(s1, s2)|$ is the length of overlap of sequences *s1* and *s2*. Given sequences *s1*, *s2* where no overlap occurs, the possibility of similarity does not exist.

Confidence

The confidence μ_{qs} for each contig is defined as a measurement of the quality of the contributing base pairs [11]. A strong signal indicates a correct read or less chance of an experimental error. Every base involved in the contig has a quality score, and the entire sequence can be a mix of low and high quality bases. The confidence of a contig is the aggregate quality score of its contributing bases. For simplicity, the sum of weighted average quality scores is the confidence of the contig. The weight can be calculated as shown in (2.2). The bases with high quality are assigned a weight of 1. The bases that are of lower quality are given weights between 0 and 1, based on the cut-off value.

$$\mu_i = \begin{cases} 1, if \quad q_i \geq \delta \\ 0, if \quad q_i = 0 \\ (q_i - min_{qs})/(max_{qs} - min_{qs}), otherwise \end{cases} \tag{2.2}$$

In (2.2), δ is the threshold as explained earlier, generally specified by the user, and min_{qs} and max_{qs} are the minimum and maximum values for quality. The minimum and maximum values are obtained from the quality scoring algorithm. The equation below describes the membership function:

$$w_{qs} = \frac{\sum_{i=0}^{n} w_i q_i}{n} \tag{2.3}$$

In (2.3), μ_{qs} is the quality score for the overall overlap region, w_i is the weight used to standardize the quality scores, n is the number of bases, and q_i is the quality score of an individual base.

Gap Penalty

Gaps refer to regions of a sequence that are missing. These are divided into three categories: Inserts, Deletes, and Replacements. Affine gap penalty can be calculated as given in Equation (2.4):

$$GapPen = GapOpening + Gaplength \times GapExtension \tag{2.4}$$

In the previous equation, *GapOpening* and *GapExtension* are scores for an opening or a continuation of a gap. The summation of (2.4) gives the entire gap penalty *GapPen(s1, s2)*. The membership function for gap penalty is given as follows:

$$\mu_{gp}(s1, s2) = \begin{cases} 1, if \;\; GapPen(s1, s2) = 0 \\ 0, if \;\; Overlap(s1, s2) \leq GapPen(s1, s2) \\ 1 - ((Overlap(s1, s2) - GapPen(s1, s2)))/(Overlap(s1, s2)), \\ \qquad otherwise \end{cases}$$

Score

The score, denoted μ_{ws}, is calculated from the numbers of matching bases, indels and replacements. The score can be calculated by using different methods. For example:

$$score = n(MatchingBP) - n(Inserts) - n(Deletes) - n(Replacements)$$

Here, n refers to the count. The fuzzy membership function for the score is defined as

$$\mu_{ws}(s1, s2) = \begin{cases} 1, if \; tscore(s1, s2) = fmbp(s1, s2) \\ 0, if \; fmbp(s1, s2) \leq 0 \\ fmbp(s1, s2)/tscore(s1, s2), otherwise \end{cases} \tag{2.5}$$

where *fmbp (s1, s2)* is the score calculated using a scoring matrix and *tscore (s1, s2)* is score of the overlap if there were no indels or replacements. Detailed explanation of fuzzy membership functions and a sample scoring matrix can be found in [24].

Fuzzy Thresholds

Minmatch

Minmatch is the minimum number of matching bases that are required between the two sequences. It is not always possible to get a perfect overlap, and some amount of inexactness is tolerated. Therefore, we would like to have a minimum match value for the overlap sequences, which has a perfect match without any

gaps. Generally, minmatch is used as a threshold to select or reject the contigs. A sigmoid membership function is used to select an optimal threshold for the minimum match required. The sigmoid function is given as

$$S(x, c) = \frac{1}{1 + e^{-(x-c)}} \tag{2.6}$$

In (2.6), x is the minmatch value selected by the user, and c is the break point that determines a transition from membership to non-membership.

Minscore

A score is calculated from the numbers of matching bases, indels and replacements as given previously in (2.5). Minscore is a threshold which specifies the minimum allowable score of the overlap. Minimum score is a commonly used parameter that sets a limit on the minimum score value that must be satisfied to accept a sequence as a match.

Aggregate Fuzzy Value

Once the fuzzy value for each of these parameters is calculated, it is combined into an function to determine the overall fuzzy value. To make a selection, this value needs to be defuzzified or converted to a crisp result. The aggregate fuzzy match value (AFV) acts as the defuzzification function. We employ the center of area (COA) defuzzification function that uses weighted average values of the fuzzy members. In a scenario of exact matching, perfect overlap can be defined as an overlap that satisfies the two thresholds, minmatch and minscore, is free of gaps, and satisfies the quality requirements. In a fuzzy system, this perfect match has a crisp value of 1. All matches that are closer to 1 than to 0 are known to be more similar. We define the fuzzy aggregate function in Equation (2.7).

$$\begin{aligned} fa(c) = \mu_{qs} w_{qs} + \mu_{ws} w_{ws} + \mu_{gp} w_{gp} + \mu_{lo} w_{lo}, \\ where : w_{qs} + w_{ws} + w_{gp} + w_{lo} = 1 \end{aligned} \tag{2.7}$$

Each of the selected parameters has a weight w associated with them. These weights can be selected by the user to control the influence of an individual factor on the assembly. For example, to achieve a stringent assembly with the least gaps, w_{gp} can be set to a higher value. To obtain longer contigs, w_{lo} can be set to a higher value. A weight can be assigned a zero value so that the factor does not influence the assembly.

$$afv = fa(c)/m \tag{2.8}$$

In (2.8), m is the number of parameters, and *fa(c)* is given in (2.7). Equations (2.7) and (2.8) give the overall fuzzy function and the aggregate fuzzy function for m parameters. The subsequences that produce the highest fuzzy value for an overlap are selected as final sequences. Depending on their position as a suffix or prefix, a new contig or consensus sequence is formed.

2.3.3 LCS Clustering

Genome sequence assembly is a rigorous task that performs comparisons of a genome with every other genome present in the population. As discussed in the background review, there have been techniques to divide the fragments into groups. These groups are intended to be small and to have high similarity between the fragments.

In this work we perform a classification based on the AFV of the LCS. The idea of grouping based on the AFV derives from the fact that sequences that satisfy the overall requirement have higher similarity. These sequences have a higher chance of forming a consensus sequence. The process named ClusFGS is described in [26]. This technique improves the performance of assembly as shown in Fig. 2.2.

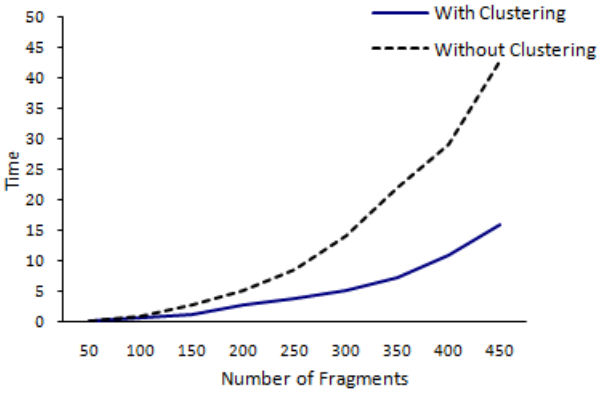


Fig. 2.2. Comparison of LCS with and without Clustering

2.3.4 Experiment and Results

The assembler was tested on artificially generated datasets and genome sequences obtained from GenBank belonging to different groups.The experiments for assembly are shown in Table 2.1. The results are compared with TIGR 2.0 [42]. The genomes used are listed as follows: (1) *The Wolbachia endosymbiont of the Drosophila melanogaster strain wMel 16S ribosomal RNA* gene containing 8,514 base pairs; (2) *Geobacillus thermodenitrificans NG80-2 plasmid pLW1071*, complete sequence, containing 57,693 base pairs; (3) *Yersinia pestis Pestoides F plasmid CD*, complete sequence, containing 71,507 base pairs; (4) *Arabidopsis thaliana genomic DNA, chromosome 3(ch3), BAC clone:F11I2*, geneid: F11I2.4. containing 36,034 base pairs; (5) *Ostreococcus tauri mitochondrion*, complete genome containing 44,237 base pairs; and, (6) *Phytophthora sojae mitochondrion*, complete genome containing 42,977 base pairs. All these genomes can be obtained from GenBank [27]. The total base was covered 4 times, *4X* of the original sequence. Each subsequence is in the range of 300-900 base pairs. In

Table 2.1. Assembly Comparisons of Different Sequences

	Percentage Genome Recovered		
Genome	MGS	TIGR	FGS
RPObc of Wolbachia genome	65%	99.6%	99.6%
Yersinia pestis Pestoides		93.9%	88.7%
Geobacillus thermodenitrificans		77.1%	91.6%
Arabidopsis thaliana ch3	56.8%	88.8%	92.135%
Ostreococcus tauri mitochondrion		77.7%	97.3%
Phytophthora sojae mitochondrion		97.7%	97.2%

Table 2.2. Table shows time for assembly and number of contigs obtained for Assembly of Sequences using FGS, the experiments were conducted on AMD Turion 64 X2 dual-core processor, with 4GB of RAM

	FGS Assembly		
Genome	Time in Sec.	No. of Sequences	No. Contigs
RPObc of Wolbachia genome	146	100	15
Geobacillus thermodenitrificans	5800	650	195
Arabidopsis thaliana ch3	466	200	61
Phytophthora sojae mitochondrion	1085	300	72

Table 2.1, MGS = Multiple Genome Sequencing using a simple LCS implementation, TIGR = TIGR Assembler 2.0, FGS = Fuzzy Genome Sequencing. Since MGS did not perform well, we did not include it in further experiments. The next experiment was to separate two species. Sequences from two organisms was taken and mixed with each other. The input data appears as if it is from a single organism. ClusFGS algorithm is performed to group sequences from the organisms, into small classes, followed by assembly. In Table 2.3, ClusFGS is the method described in Subsect. 2.3.3 and is a modified version of FGS. Misclassification refers to the length of overall subsequences from genome 1 that were assembled incorrectly with contigs of genome 2. The results obtained in Table 2.1 from assembling the genome projects showed a high percentage of the genome recovered while using FGS and TIGR. Some of the small differences in results could be due to different thresholds being used. Preliminary results from Table 2.3 show that fuzzy classification was successful in grouping these two classes separately. The clustering classified the data into two groups without any misclassification. The clustering technique is linear, and hence, can make the

Table 2.3. Clustering for Two Organisms with ClusFGS

Genome	Percentage Recovered	Miss-Classifications
P. sojae mitochondrion	61%	0%
G. thermodenitrificans	61.1%	0%

assembly much faster. At this stage ClusFGS cannot recover a higher percentage of the genome because comparisons are done within a class. The performance is limited by factors such as random selection of the seeds, no interaction between classes such as reassignment of sequences, and smaller classes not being merged. This classification with some of it drawbacks is the inspiration for the new work that is presented in Sect. 2.4.

2.4 Fuzzy Classifier to Taxonomically Group DNA Fragments within a Metagenome

The metagenomic approach makes the acquisition of genomic fragments easier; nevertheless, the approach suffers from limitations. Recall from Sect. 2.2 that the diverse genomes acquired together may need to be separated and assembled to make meaningful conclusions. Taxonomical classification of genomic fragments is a vital problem in metagenomic approach. Because these microorganisms come from the same community, their characteristics are similar. Nevertheless, closely related bacteria can contain remarkable genomic diversity [49]. These differences can be found by analysis of features of the DNA, which we refer as DNA signature.

Pre-assembly grouping of metagenomic fragments into phylogenetic classes can lead to faster and more robust assembly by reducing the search space required to find adjacent fragment pairs, because DNA from the same organism should be classified into the same taxonomic group. The DNA signatures chosen are GC content, and tri- and tetra-nucleotide frequencies. The proposed method uses a fuzzy classifier and extracts signatures from given sequences and uses them as a feature set. The technique is verified with artificial shotgun sequences to measure correctness. The main purpose is to classify fragments of a community, which is depicted by classification of an acid mine drainage (AMD) environmental genome.

Even though studies have successfully taxonomically differentiated full genomes or fragments of sizes greater than 1,000 base pairs [22, 43, 51], there is a lack of availability of applications that classify shorter (500-900 base pairs) shotgun fragments. The proposed approach is designed with a goal of classifying shotgun fragments. Earlier techniques have focused on using a single signature for classification [51]. A combination of different signatures is proposed for the classification.

2.4.1 Background

Separation of domain-specific genomic fragments and reconstruction is a complex process that involves identification of certain features exhibited by entire taxonomic groups. These features are used to group the metagenomic sample into classes. The following subsection describes the DNA signatures that are employed in identification or classification of fragments.

DNA Signatures

Phylogenetically related groups of sequences show similar nucleotide frequencies either because of convergence or because they were inherited from a common ancestor [8]. For example, a study conducted on *E-coli* revealed a nonrandom utilization of codon pairs [16]. Some of the most frequent codon pairs found were: CTGGCG, CTGGCC, CTGGCA, CTGGAC, AACCCG, CTGGAA. This study and others reveal that there is a nonrandom over-representation and under-representation of certain codon pairs within a species. Oligonucleotide frequency studies with short x-x bases have reported tendencies of under- and over-representation n Xmers [5]. This study brought to attention that certain oligonucleotides are rarely observed in certain species while certain other oligonucleotides have shown their dominance in a particular species. This also shows that nucleotide composition contains bias.

The key to the classification of genomes is the presence of atterns in a sequence. Recall from Sect. 2.2.3, that these patterns can be specific to certain organism group. Thus, identification of these patterns can lead to the discovery of the phylogeny of a group. Moreover, the patterns can be used as signatures to distinguish one species from another. We now move our discussion to the two groups of signatures that will be utilized.

The first signature is based on GC content present in the genome. GC content is found to be variable with different organisms; this variation is viewed to be the result of variation in selection or bias in mutation [3]. For example, coding regions within a genome code for genes and are less divergent within populations. enes represent characteristics of an organism: the physical development and phenotype of organisms can be thought of as a product of genes interacting with each other and with the environment [28]. Studies have shown that the length of the coding sequence is directly proportional to higher GC content [29], thus showing a strong correlation between GC content and gene properties. The pre-assembly of a well-known metagenomic dataset from acid mine drainage was performed by binning the fragments by their GC content [46].

The second signature that were investigated were the oligonucleotide frequencies. Nucleotide frequencies are generally taken from a group of two, three, four, five, or six nucleotides. These are known as di-, tri-, tetra-, penta- and dicodon nucleotide frequencies, respectively. These prefixes indicate the presence or absence of certain words in a genome that have been used to separate certain species. Evaluation of frequencies of fragments and their correlations based on taxonomy was performed by Teeling, *et al.* [43]. In this paper it was shown that GC content is not sufficient for separating species and tetra-nucleotide frequencies showed better differentiation of species for fragments of size 40,000 base pairs. A grouping based on nucleotide frequencies resembles the phylogenetic grouping of the representative organisms [33]. In another approach, differentiation of bacterial genomes was performed using statistical approaches for structural analysis of nucleotide sequences [35].

Frequencies of larger word sizes such as tetra, penta, and hexa are considered more reliable. But obtaining enough frequencies for larger words is difficult and

may not give statistically relevant results for shorter fragments. There are a total of 4,096 dicodons. A sequence of length 10,000 bp contains 1,665 dicodons because this number is less than 4,096, the sequence cannot cover the 4,096 dicodons. The same sample contains 3,332 tri-nucleotide frequencies, that can easily cover the 64 tri-nucleotides. Therefore, it is better to use tri-nucleotide frequencies in cases of fragment classification.

Even though studies have successfully taxonomically differentiated full genomes or fragments of sizes greater than 1,000 base pairs [22, 43], there is a lack of availability of applications that classify shorter (500-900bp) shotgun fragments. Our approach is designed with a goal of classifying shotgun fragments. Earlier techniques have also focused on using a single signature for classification. We propose using a combination of different signature patterns.

Clustering

K-means is an unsupervised learning algorithm to group objects into categories. The simplest K-means algorithm places N objects into K classes by using the minimum distance from the center of K to each object. In the simple K-means approach, K is fixed a priori. Clustering problems generally derive some kind of similarity between groups of objects. K-means clustering is a simple and fast approach to achieve a grouping for data. Due to its simple method of using feature vectors as seeds and the arithmetic mean as the center for the clusters, the K-means algorithm suffers from drawbacks. The simple K-means algorithm could not guarantee convergence. A modified K-means was developed that uses a weighted fuzzy average instead of the mean to get new cluster centers. Using a fuzzy weighted average instead of a simple mean improved K-means and also leads to convergence [21]. In this research, a modification of the fuzzy K-means algorithm with fuzzy weighted averages is used for fragment clustering. The algorithm is described in the next subsection.

2.4.2 An Overview of Our Algorithm

Fragment classification divides entire datasets into smaller categories. The classes represent two significant properties: (1) they contain fragments belonging to the same group or species present in the metagenomic data set, and (2) they have continuity and can represent local regions of the genome. The first step to classification is the identification of the signatures for each fragment. After the signatures are extracted the feature vector is initialized, and the K-means algorithm is run to create classes. The operations carried out is be described next.

GC Content

GC content is expressed as the percentage of G and C present in the fragment and is calculated as follows:

$$\frac{C+G}{A+C+G+T} \times 100$$

Certain factors need to be considered when using GC content. GC content is known to be more influential in coding regions. Shotgun fragments of metagenomes do not contain information that reveals directly whether a certain fragment contains coding regions or the percentage of fragment region that can code for a gene. Analysis of GC content revealed that it is not sufficient to obtain a classification when closely related species are present in the datasets. Thus, advanced signatures are required to obtain a good separation of groups within a metagenome.

Nucleotide Frequencies Using Markov Chain Model

Markovian models have been used in fields such as statistics, physics and queuing theory. Markov chain predictors have also been used to predict coding regions, thus finding genes. The simplest chain is the zero-order Markov chain which can be estimated from the frequencies of the individual nucleotides A, C, G, and T. The approach used to estimate the zero-order Markov chain is shown below. Consider the sequence GGATCCC, the nucleotide frequency is given by:

$$p(GGATCC) = p(G)p(G)p(A)p(T)p(C)p(C)$$

Higher order Markov chains can also be constructed using only the previous state frequencies. A maximal-order Markov chain removes biases from all the previous states and is dependent on only the past state. frequencies and tetra-nucleotide frequencies. The tri-nucleotide and tetra-nucleotide frequencies can be calculated using a maximal-order Markov chain. Expected values are directly calculated from the observed values as shown in (2.9). In (2.9) and (2.10), O refers to the observed values, E is the expected value, and N_i refers to a nucleic acid base pair.

$$E(N_1N_2N_3) = \frac{O(N_1N_2)O(N_2N_3)}{O(N_2)} \tag{2.9}$$

$$E(N_1N_2N_3N_4) = \frac{O(N_1N_2N_3)O(N_2N_3N_4)}{O(N_2N_3)} \tag{2.10}$$

Fuzzy K-Means Clustering

Clustering for a metagenome assembly problem has a two fold purpose: to divide the space for performance improvement and to group fragments into classes such that each class has fragments from one group. The K-means algorithm uses a set of unlabeled feature vectors and classifies them into K classes. From the set of feature vectors K of them are randomly selected as initial seeds. The feature vectors are assigned to the closest seed. The mean of features belonging to a class is taken as the new center.

Given N sequences, such that $S = \{C\}^i$, where $C = \{A,\ C,\ G,\ T\}$. We randomly select K sequences as the initial seeds, where K is less than the number of sequences N. The nucleotide frequencies and GC content for all sequences are calculated. These frequencies form the p features to be used in classification.

The sequence is assigned to the class that has the highest fuzzy similarity. The fuzzy similarity is calculated using a weighted fuzzy average (WFA). Let

$\{x_1, ..., x_P\}$ be a set of P real numbers. The weighted fuzzy average is using the weight w_p for x_p is given as:

$$\mu^r = \sum_{p=1,P} w_p^{(r)} x_p, \; r \; = 0, 1, 2, \ldots$$

Here, x is the parameter or feature and p the number of features. The number of the iteration is given as r. The mean is obtained for all the K initial classes. The next step is to assign features to each of the classes. A feature is assigned to the closest class by computing the distance of a feature from each of the classes. Given i=0,. . . ,N and j=0,. . . ,k, the distance $d_{i,j}$ for each cluster can be calculated as follows:

$$d_{i,j} = max(\mu_j^r), for \; all \; j = 0, \ldots, k$$

Thus feature vectors are assigned to a class. Since a large number of classes were created initially, empty or small classes are eliminated. Classes that are close to each other are merged to form one class. This process is repeated until convergence by replacing the initial mean with the WFA, and feature vectors are reassigned by computing the distance. In the next subsection we show the results obtained by classification and describe the genomes used to test the approach.

2.4.3 Clustering *via* Feature Extraction

Artificial Metagenome

To asses the performance of fuzzy clustering on genomic sequences, experiments on artificial data were performed. In the first experiment, two genomes from different phylogenetic types are used for the first test case. These fragments are mixed with each other. Table 2.4 shows the results obtained after classifying these two samples. In Tables 2.4 and 2.6, GC refers to clustering with GC content, T_z refers to tri-nucleotide and TR_z refers to tetra-nucleotide frequencies using zero-order Markov chains. TR_m refers to tetra-nucleotide frequencies using a maximal-order Markov chain, T_m indicates the tri-nucleotide frequencies using a maximal-order Markov chain. Combinations of different signatures are shown by hyphenating individual frequencies. A value of NA indicates that the signatures could not separate the fragments into groups and all the data was placed into one class. In the second experiment, the dataset that was described in Sect. 2.3.4 was used. For this experiment we conducted not only classification but also assembly of the sequence using signature-based classification. Recall results from Table 2.3, that classified genome fragments using an LCS-based approach. The results of our classification and assembly are shown in Table 2.5. These results indicate improvement in assembly using the signature-based method, even though there are few misclassifications. The reason for the misclassifications is that ClusFGS classifies based on LCS, which considers the entire sequence for classification. Whereas signature-based classification uses signatures without creating an overlap, this also makes the approach much faster than ClusFGS.

Table 2.4. Separating 500 Fragments Belonging to Two Organisms Using Different Signatures

	# Fragments classified incorrectly		
Signature	Genome 1	Genome 2	Total %
GC	39	1	0.08
T_z	NA	NA	NA
TR_z	NA	NA	NA
$GC - T_z - TR_z$	18	32	0.1
T_m	7	0	0.02
TR_m	15	0	0.03
T_m- TR_m	11	3	0.028
$GC - T_m - TR_m$	5	1	0.012

Table 2.5. Clustering and Assembly of 800 Artificial Metagenome fragments

Genome	Percentage Recovered	Miss-Classifications
P. sojae mitochondrion	82.90%	0%
G. thermodenitrificans	94.124%	1.6%

The Acid Mine Drainage Metagenome

The AMD metagenome was obtained from Richmond Mine at Iron Mountain, CA [46]. The acid mine drainage environmental genome was shown to contain 2 major groups. We use shotgun sequences of two genomes of AMD, namely *Leptospirillum sp. Group II* (Lepto) environmental sequence and *Ferroplasma sp. Type II*(Ferrop. Type II) environmental sequence. These sets are 960,150 and 1,317,076 nucleotide base pairs respectively. The first group belongs to the bacterial genus *Leptospirillum*; the second one is an archea from the genus *ferroplasma*. Shotgun sequences of average size 700 base pairs were generated from these genomes. Fig. 2.3 depicts the classification results on AMD data, using GC content. A set of 3,000 samples was used for the display. The tests were conducted successfully for all 5 sub-groups in AMD.

The results of classification using the modified K-means approach using DNA signatures is given in Table 2.6. It compares the classification results for the two AMD genomes. Classification was performed using different combinations of signatures, and the results are displayed in Table 2.6. The final classification resulted in two groups, one with fragments from *Lepto* and another with *Ferrop Type II* fragments respectively. The results indicate that frequencies obtained using maximal-order Markov chain created the better classification than zero-order Markov chain. A combination of different signatures also resulted in fewer misclassifications.

Taxonomy is a method of classifying organisms into types and further classifying types into subtypes to form a hierarchical structure. All species are classified into hierachical groups starting with *domain*, *kingdom*, *phylum*, *class*, *order*,

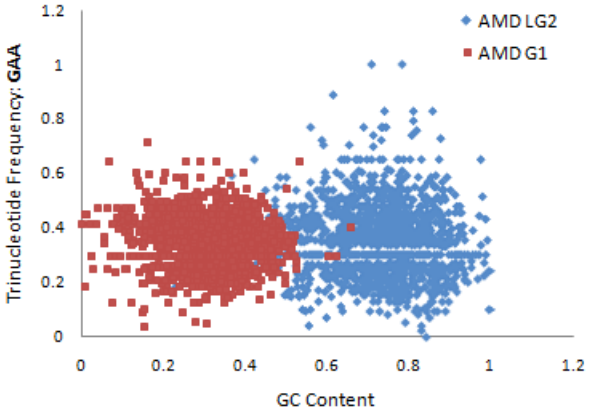


Fig. 2.3. Classification using GC Content and Nucleotide Frequency for Shotgun Sequence Fragments obtained from AMD G1 and AMD LG2

family, *genus* and *species*. Organisms that belong to different domains can have genome sequences that are different. But as we go down the hierarchy the similarities increase, therefore organisms that belong to same species are highly similar. As similarities between organisms increase it becomes difficult to cluster them. A study on the classification of fragments to identify the accuracy of the classifier can be found in [24]. Analysis of certain pairs also shows that there is over- and under-representation of certain oligonucleotide words. The results of classification indicate that at higher ranks in the taxonomy the classifier works well and the classification gradually decreases after which there is sharp increase in miss-classifications. Advanced signatures or supervised clustering can be a potential approach for organisms that are more similar.

Table 2.6. Separating 20,000 Fragments from AMD into Two Classes Using Different Signatures

	# Fragments classified incorrectly		
Signature	Lepto.	Ferrop. Type II	Total %
GC	500	27	0.026
T_z	NA	NA	NA
TR_z	NA	NA	NA
$GC - T_z - TR_z$	640	27	0.033
T_m	147	16	0.0081
TR_m	170	6	0.0088
T_m- TR_m	127	10	0.0068
$GC - T_m - TR_m$	129	11	0.007

2.5 Conclusions and Future Work

This body of work contributes an effective framework for assembly of sequences using fuzzy logic. The work was initiated to create an assembler that can work on metagenome fragments without pre-processing. The fuzzy assembly process can successfully assemble sequences. The functions proposed can be easily adapted in other assembly methods or techniques.

This classifier is enhanced to use DNA signatures to perform a phylogenetic classification. A fuzzy clustering algorithm is proposed to classify shotgun genome fragments into taxonomical classes. We classified fragments using different signatures and combination of signatures. We also tested the AMD metagenome and classified it into two groups of bacteria and archea. Using combination of DNA signatures also showed good classification. Results were obtained for different types of genomes sequences, thus testing a wide range of input genomes. Prior to this work, classification was performed on full genomes or fragments that were longer than 1000bp. This work shows that fragments of smaller length can also be classified into groups. We propose an unsupervised classification that requires, no training or identification of important nucleotides. A known limitation of the classification technique is that the classes have to be set by the user. If the classes are not set, the K-means algorithm determines final groups. The algorithm creates classes that are compact rather than classes that are large and dispersed. Thus fragments from one genome, can be present in more than one class, ensuring classes with minimal or no misclassifications. The technique can be improved by application of validity measures, using marker regions to identify and create groups that can represent a number of genomes within the sample.

This work opens a question of using an adaptive assembler that can adapt itself to the input to generate the best possible assembly. The concept of adaptive assembler is dependent on two factors: statistical analysis of data and the best approximation of the parameters. The assembly can be further improved by data reduction before assembly making it possible to run larger data sets at faster speeds. A parallel version of the assembler can be found faster assembly in [24]. The classification proposed can also be enhanced by generating signatures that are different from each other rather than selecting random signatures. The results indicate that we are able to group shotgun sequences from their frequencies and GC Content. Analysis of the DNA signatures can be done to find the best discriminatory pairs, enabling selection of features that suit the dataset best rather than using all available frequencies.

References

1. Baxevanis, A.D., Ouellette, B.F.F.: Bioinformatics: A practical guide to the analysis of genes and proteins, 1st edn. John Wiley, Chichester (2005)
2. Beja, O., Suzuki, M.T., Koonin, E.V., Aravind, L., Hadd, A., Nguyen, L.P., Villacorta, R., Amjadi, M., Garrigues, C., Jovanovich, S.B., Feldman, R.A., DeLong, E.F.: Construction and analysis of bacterial artificial chromosome libraries from a marine microbial assemblage. Environmental Microbiology 2, 516–529 (2000)

3. Birdsell, J.A.: Integrating genomics, bioinformatics and classical genetics to study the effects of recombination on genome evolution. Molocular Biology Evolution 19, 1181–1197 (2002)
4. Brown, T.A.: Genomes, 3rd edn. Garland Science (2006)
5. Burge, C., Campbell, A.M., Karlin, S.: Over- and under-representation of short oligonucleotides in DNA sequences. Proceedings National Acaddemy of Science USA 89(4), 1358–1362 (1992)
6. Chen, K., Pachter, L.: Bioinformatics for whole-genome shotgun sequencing of microbial communities. PLoS Computational Biology 1, 106–112 (2005)
7. Choudhuri, S.: The path from nuclein to human genome: A brief history of DNA with a note on human genome sequencing and its impact on future research in biology. Bulletin of Science Technology Society 23, 360–367 (2003)
8. Conant, G.C., Lewis, P.O.: Effects of nucleotide composition bias on the success of the parsimony criterion in phylogenetic inference. Molecular Biology Evolution 18, 1024–1033 (2001)
9. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to algorithms, 2nd edn., pp. 313–319. McGraw-Hill, New York (2001)
10. Edmund, P.: A history of genome sequencing, Tech. report, Yale University Bioinformatics (2001)
11. Ewing, B., Green, P.: Basecalling of automated sequencer traces using phred. ii. error probabilities. Genome Research 8, 186–194 (1998)
12. Fleischmann, R., Adams, M., White, O., Clayton, R., Kirkness, E., Kerlavage, A., Bult, C., Tomb, J., Dougherty, B., Merrick, J.: Whole-genome random sequencing and assembly of Haemophilus Influenzae Rd. Science 269(5223), 496–512 (1995)
13. Gasch, A.P., Eisen, M.B.: Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. Genome Biology 3(11), 1–22 (2002)
14. Gene, M.: Whole-genome DNA sequencing. IEEE Computational Engineering and Science 1, 33–43 (1999)
15. Green, P.: Documentation for phrap. Genome Center, University of Washington (2006)
16. Gutman, G.A., Hatfield, G.W.: Nonrandom utilization of codon pairs in Escherichia coli. Proceedings National Academy of Science USA 86, 3699–3703 (1989)
17. Huang, X., Madan, A.: CAP3: A DNA sequence assembly program. Genome Research 9(9), 868–877 (1999)
18. Hugenholtz, P.: Exploring prokaryotic diversity in the genomic era. Genome Biology 3, reviews0003.1–reviews0003.8 (2002)
19. Karlin, S., Ladunga, I., Blaisdell, B.E.: Heterogeneity of genomes: Measures and values. Proceedings National Academy of Science USA 91, 12837–12841 (1994)
20. Kececioglu, J.D., Myers, E.W.: Combinatorial algorithms for DNA sequence assembly. Algorithmica 13, 7–51 (1995)
21. Looney, C.G.: Interactive clustering and merging with a new fuzzy expected value. Pattern Recognition 35, 2413–2423 (2002)
22. McHardy, A.C., Martín, H.G., Tsirigos, A., Hugenholtz, P., Rigoutsos, I.: Accurate phylogenetic classification of variable-length DNA fragments. Nature Methods 4(1), 63–72 (2007)
23. Mongodin, E., Emerson, J., Nelson, K.: Microbial metagenomics. Genome Biology 6(10), 347 (2005)
24. Nasser, S.: Fuzzy sequence classification and assembly of environmental genomes, Ph.D. thesis, University of Nevada Reno (2008)

25. Nasser, S., Vert, G., Nicolescu, M., Murray, A.: Multiple sequence alignment using fuzzy logic. In: Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, Honolulu, Hawaii, vol. 7, pp. 304–311 (2007)
26. Nasser, S., Vert, G.L., Breland, A., Nicolescu, M.: Fuzzy classification of genome sequences prior to assembly based on similarity measures. In: North American Fuzzy Information Processing Society, pp. 354–359 (2007)
27. NCBI, National center for biotechnology information, NIH (2007), `http://www.ncbi.nlm.nih.gov/`
28. Nowak, M.A.: Evolutionary dynamics: Exploring the equations of life, 1st edn. Belknap Press (October 2006)
29. Oliver, J.L., Marín, A.: A relationship between GC content and coding-sequence length. Journal of Molecular Evolution 43(3), 216–223 (2004)
30. Otu, H.H., Sayood, K.: A divide-and-conquer approach to fragment assembly. Bioinformatics 19(1), 22–29 (2003)
31. Peltola, H., Soderlund, H., Ukkonen, E.: Seqaid: A DNA sequence assembling program based on a mathematical model. Nucleic Acids Research 21(1), 307–321 (1984)
32. Pop, M., Salzberg, S.L., Shumway, M.: Genome sequence assembly: Algorithms and issues. IEEE Computer 35(7), 47–54 (2002)
33. Pride, D.T., Meinersmann, R.J., Wassenaar, T.M., Blaser, M.J.: Evolutionary implications of microbial genome tetranucleotide frequency biases. Genome Research 13(2), 145–158 (2003)
34. Rappe, M., Giovannoni, S.: The uncultured microbial majority. Annual Reviews Microbiology 57, 369–394 (2003)
35. Reva, O., Tümmler, B.: Differentiation of regions with atypical oligonucleotide composition in bacterial genomes. BMC Bioinformatics 6(1), 251 (2005)
36. Rondon, M.R., August, P.R., Bettermann, A.D., Bradly, S.F., Grossman, T.H., Liles, M.R., Loiacono, K.A., Lynch, B.A., MacNeil, I.A., Minor, C., Tiong, C.L., Gilman, M., Osburne, M.S., Clardy, J., Handelsman, J., Goodman, R.M.: Cloning the soil metagenome: a strategy for accessing the genetic and functional diversity of uncultured microorgansims. Applications Environmental Microbiology 66, 2541–2547 (2000)
37. Sadegh-Zadeh, K.: Fuzzy genomes. Artificial Intelligent Medicine 18(1), 1–28 (2000)
38. Sanger, F., Coulson, A.R., Hong, G.F., Hill, D.F., Petersen, G.B.: Nucleotide sequence of Bacteriophage Lambda DNA. Journal Molecular Biology 162(4), 729–773 (1982)
39. Sanger, F., Nicklen, S., Coulson, A.R.: DNA sequencing with chain-terminating inhibitors. Proceedings National Academy of Science USA 74(12), 5463–5467 (1977)
40. Smith, T., Waterman, M.: Identification of common molecular subsequences. Journal of Molecular Biology 147, 195–197 (1981)
41. Stein, J.L., Marsh, T.L., Wu, K.Y., Shizuya, H., DeLong, E.F.: Characterization of uncultivated prokaryotes: isolation and analysis of a 40-kilobase-pair genome fragment from a planktonic marine archaeon. Journal of Bacteriology 178, 591–599 (1996)
42. Sutton, G., White, O., Adams, M., Kerlavage, A.: TIGR Assembler: A new tool for assembling large shotgun sequencing projects. Genome Science & Technology 1, 9–19 (1995)

43. Teeling, H., Meyerdierks, A., Bauer, M., Amann, R., Glockner, F.: Application of tetranucleotide frequencies for the assignment of genomic fragments. Environmental Microbiology 6, 938–947 (2004)
44. Teeling, H., Waldmann, J., Lombardot, T., Bauer, M., Glockner, F.O.: TETRA: A web-service and a stand-alone program for the analysis and comparison of tetranucleotide usage patterns in dna sequences. BMC Bioinformatics 5, 163 (2004)
45. Turnbaugh, P.J., Ley, R.E., Mahowald, M.A., Magrini, V., Mardis, E.R., Gordon, J.I.: An obesity–associated gut microbiome with increased capacity for energy harvest. Nature 444(7122), 1009–1010 (2006)
46. Tyson, G.W., Chapman, J., Hugenholtz, P., Allen, E.E., Ram, R.J., Richardson, P.M., Solovyev, V.V., Rubin, E.M., Rokhsar, D.S., Banfield, J.F.: Community structure and metabolism through reconstruction of microbial genomes from the environment. Nature 428, 37–43 (2004)
47. Ukkonen, E.: On-line construction of suffix trees. Algorithmica 14(3), 249–260 (1995)
48. Venter, J.C., Remington, K., Heidelberg, J.F., Halpern, A.L., Rusch, D., Eisen, J.A., Wu, D., Paulsen, I., Nelson, K.E., Nelson, W., Fouts, D.E., Levy, S., Knap, A.H., Lomas, M.W., Nealson, K., White, O., Peterson, J., Hoffman, J., Parsons, R., Baden-Tillson, H., Pfannkoch, C., Rogers, Y.-H., Smith, H.O.: Environmental genome shotgun sequencing of the sargasso sea. Science 304, 66–74 (2004)
49. Welch, R.A., Burland, V., Plunkett, G., Redford, P., Roesch, P., Rasko, D., Buckles, E.L., Liou, S.R., Boutin, A., Hackett, J., Stroud, D., Mayhew, G.F., Rose, D.J., Zhou, S., Schwartz, D.C., Perna, N.T., Mobley, H.L., Donnenberg, M.S., Blattner, F.R.: Extensive mosaic structure revealed by the complete genome sequence of uropathogenic Escherichia coli. Proceedings National Academy of Science USA 99(26), 17020–17024 (2002)
50. Wong, L.: The practical bioinformatician, 1st edn. World Scientific Publishing Company, Singapore (2004)
51. Woyke, T., Teeling, H., Ivanova, N.N., Huntemann, M., Richter, M., Gloeckner, F.O., Boffelli, D., Anderson, I.J., Barry, K.W., Shapiro, H.J., Szeto, E., Kyrpides, N.C., Mussmann, M., Amann, R., Bergin, C., Ruehland, C., Rubin, E.M., Dubilier, N.: Symbiosis insights through metagenomic analysis of a microbial consortium. Nature 443(7114), 925–927 (2006)
52. Xu, D., Bondugula, R., Popescu, M., Keller, J.: Bioinformatics and fuzzy logic. In: IEEE International Conference on Fuzzy Systems, Vancouver, BC, pp. 817–824 (2006)
53. Zadeh, L.A.: Fuzzy logic and approximate reasoning. Synthese 30 (1975)
54. Zhang, Z., Schwartz, S., Wagner, L., Miller, W.: A greedy algorithm for aligning DNA sequences. Journal of Computational Biology 7, 203–214 (2000)

3

A Hybrid Promoter Analysis Methodology for Prokaryotic Genomes

Oscar Harari[1], Luis Herrera[1], and Igor Zwir[1,2]

[1] Dept. Computer Science and Artificial Intelligence, University of Granada, E-18071, Spain
oharari@decsai.ugr.es
[2] Howard Hughes Medical Institute, Department of Molecular Microbiology, Washington University School of Medicine, St. Louis, MO 63110-1093, USA
zwir@borcimwustl.edu

Summary. One of the biggest challenges in genomics is the elucidation of the design principles controlling gene expression. Current approaches examine promoter sequences for particular features, such as the presence of binding sites for a transcriptional regulator, and identify recurrent relationships among these features termed network motifs. To define the expression dynamics of a group of genes, the strength of the connections in a network must be specified, and these are determined by the *cis*-promoter features participating in the regulation. Approaches that homogenize features among promoters (e.g., relying on consensuses to describe the various promoter features) and even across species hamper the discovery of the key differences that distinguish promoters that are co-regulated by the same transcriptional regulator. Thus, we have developed a model-based approach to analyze proteobacterial genomes for promoter features that is specifically designed to account for the variability in sequence, location and topology intrinsic to differential gene expression. We applied our method to characterize network motifs controlled by the PhoP/PhoQ regulatory system of *Escherichia coli* and *Salmonella enterica* serovar Typhimurium. We identify key features that enable the PhoP protein to produce distinct kinetic patterns in target genes, which could not have been uncovered just by inspecting network motifs.

3.1 Introduction

Whole genome sequences and genome-wide gene expression patterns (usually in the form of microarray data) provide the raw material for the characterization and understanding of transcription regulatory networks. These networks can be represented as directed graphs in which a node stands for a gene (or an operon in the case of bacteria) and an edge symbolizes a direct transcriptional interaction. Recurrent patterns of interactions, termed network motifs, occur far more often than in randomized networks, forming elementary building blocks that carry out key functions. This is a convenient representation of the topology of a set of regulatory Boolean (i.e. ON-OFF) networks, in which each gene is either fully expressed or not expressed at all, or that it has a binding site for a transcriptional regulator or lacks such a site. However, this approach has serious limitations because

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 45–65.
springerlink.com

most genes are not expressed in a simple Boolean fashion. Indeed, genes that are co-regulated by the same transcription factor are often differently expressed with characteristic expression levels and kinetics. Therefore, a deeper understanding of regulatory networks demands the identification of the key features used by a transcriptional regulator to differentially control genes that display distinct behaviors despite belonging to networks with identical motifs.

The identification of the promoter features that determine the distinct expression behavior of co-regulated genes is a challenging task because: first, there are difficulties in discerning the sequence elements relevant to differential expression patterns (e.g., the binding sites for transcriptional regulators and RNA polymerase) from a background of variable DNA sequences that do not play a direct role in gene regulation [3, 27]. Second, the sequences recognized by a transcription factor may differ from promoter to promoter within and between genomes and may be located at various distances from other *cis*-acting features in different promoters [51, 53]. Third, similar expression patterns can be generated from different or a mixture of multiple underlying features, thus, making it more difficult to discern the causes of analogous regulatory effects.

In this study, we present a method specifically aimed at handling the variability in sequence, location and topology that characterize gene transcription. Instead of using an overall consensus model for a feature, where important differences are often concealed because of intrinsic averaging operations between promoters and even across species, we decompose a feature into a family of models or building blocks. This approach maximizes the sensitivity of detecting those instances that weakly resemble a consensus (e.g., binding site sequences) without decreasing the specificity. In addition, features are considered using fuzzy assignments, which allow us to encode how well a particular sequence matches each of the multiple models for a given promoter feature. Individual features are then linked into more informative composite models that can be used to explain the kinetic expression behavior of genes.

We applied our method to analyze promoters controlled by the PhoP/PhoQ regulatory system of *Escherichia coli* and *Salmonella enterica* serovar Typhimurium. This system responds to the same inducing signal (i.e. low Mg2+) in both species [53, 11, 32, 45]. Moreover, the *E. coli phoP* gene could complement a *Salmonella phoP* mutant [16]. The DNA-binding PhoP protein appears to recognize a tandem repeat sequence separated by 5 bp [53, 11, 32], consistent with being a dimer [35]. The PhoP/PhoQ system is an excellent test case because it controls the expression of a large number of genes, amounting to ca. 3% of the genes in the case of *Salmonella* [22]. Furthermore, the PhoP/PhoQ regulon has been shown to employ a variety of network motifs including the single-input module (Figure 3.1(A)), the multi-input module (Figure 3.1(B)), the bi-fan (Figure 3.1(C)), the chained (Figure 3.1(D)), and the feedforward loop (Figure 3.1(E)) [22, 34, 24]. Our analysis uncovered the salient features that distinguish genes co-regulated by PhoP belonging to similar networks. Gene transcription measurements provided experimental support for the investigated predictions.

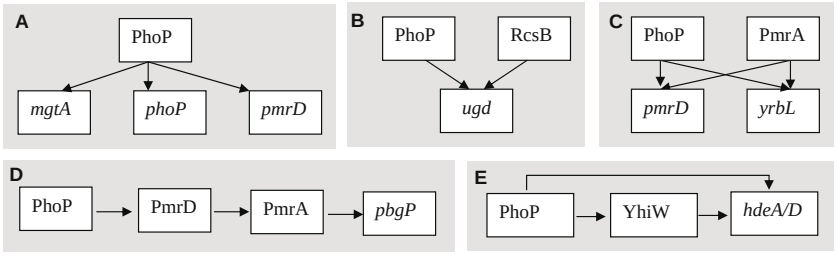


Fig. 3.1. The PhoP/PhoQ system employs a variety of network motifs to regulate gene transcription. (**A**) In the single-input module, PhoP as a single transcription factor regulates a set of genes (i.e. *mgtA*, *phoP* and *pmrD*). (**B**) In the multi-input module, two or more transcription factors (e.g., PhoP and RcsB) regulate a target gene (i.e. *ugd*). (**C**) In the bi-fan module, a set of genes (i.e. *pmrD* and *yrbL*) are each regulated by a combination of transcription factors (i.e. PhoP and PmrA). (**D**) In the chained motif, genes are regulated in an ordered cascade. (**E**) In the feedforward loop, a transcription factor (i.e. PhoP) regulates the expression of a second transcription factor (i.e. YhiW), and both jointly regulate one or more genes (i.e. *hdeA/D*).

3.2 Materials and Methods

Our method consists of three phases: first, encoding the available information into preliminary model-based features, which includes identifying *cis*-features from DNA sequences and information from available databases; performing initial modeling of each individual feature, allowing the process of multiple occurrences of a feature and using relaxed thresholds and permitting missing values. A *model-based* feature is generated by the identification of a feature in a subset of observations (F) in the dataset, based on measuring the degree of match (Q) between an observation and a model, or a family of models (M=$\{M_\alpha\}$), at some degree (α) defined in a unit-interval scale (i.e., fuzzy values, $Q(F, M_\alpha)$) [38, 54]. Second, grouping the results into subsets, thus, decomposing the preliminary models into a family of models or building blocks by using fuzzy clustering. Third, composing the building blocks by either combining the same or different types of features by using fuzzy logic expressions. And fourth, describing new promoters using the resulting models.

Network Motifs

In theory, the term "network motifs" is related to a statistical significant subgraph; however, in practice, they are treated as an over represented subgraph (see [12, 6, 5]). For example, a motif termed "single input motif " of three/four nodes in the *E. coli* [39] (e.g., mfinder1.2 p-value < 34.7+-8.5) or *Saccharomyces cerevisiae* network [20] is not recognized as significant, while the only motif that exceeds the standard threshold is the "feed forward motif".

Binding Site Submotifs and Orientation

(1) We built an initial model for the PhoP binding site by learning a position weight matrix [18] (*E-value* < 10E-12) based on the upstream sequences of genes corresponding to the training set of the *E. coli* and *Salmonella* genomes (Table S1). **(2)** We searched the intergenic regions of the genes in both orientations, using low thresholds corresponding to two standard deviations below the mean score obtained with the initial model [36]. Multiple PhoP binding site candidates were allowed in a given promoter operator region. **(3)** After transforming nucleotides into dummy variables [12], we grouped sequences matching the PhoP position weight matrix using the fuzzy C-means clustering method with the Xie-Beni validity index (see below) to estimate the number of clusters [6, 5]. **(4)** We built models for these clusters using position weight matrices (*E-value* < 10E-22) and searched the *E. coli* and *Salmonella* genomes to characterize each gene according to its similarity to each model as a fuzzy partition (Figure S1 and S2).

Performance. To evaluate the ability of the resulting models to describe PhoP-regulated promoters, we extended the dataset by including 772 promoters (RegulonDB V3.1 database [39]) that are regulated by transcription factors other than PhoP (see "Search known transcription factor motifs" [19]), by selecting the promoter region corresponding to the respective transcription factor binding site 10 bp. We considered the compiled list of PhoP regulated genes as true positive examples (Table S1) and the binding sites of other transcriptional regulators as true negative examples to evaluate the performance of the submotif feature. We used a leave-one-out crossvalidation process (Crossvalind, Matlab r2006a), which is appropriate for reduced datasets, as a procedure to estimate the variance error on the training set (correct test estimation of 94% vs. 75% between submotifs and single position weight matrices, respectively). Then, each matrix threshold has been optimized for classification purposes by using the correlation coefficient measurement (see below) based on the extended dataset (Table S2). (See the complete evaluation of genomes online [19]). We found that the PhoP-binding site model increases its sensitivity from 66% to 91% when submotifs are used instead of a single consensus, while its specificity went from 98% to 97% (correlation coefficient 73% vs. 87%). We also obtained substantial improvements for other transcription factors from RegulonDB. For example, by considering the CRP regulator, we used 130 promoters regulated by this protein in RegulonDB as the true positive values and 642 regulated by other proteins than CRP as negative examples. We found that the sensitivity of the CRP model for binding sites increases from 29% to 50%, by using submotifs instead of a single consensus, while the specificity remains the same at 98% (correlation coefficient 39% vs. 62%). Overall, by considering transcription factors with more than ten reported binding sequences in the RegulonDB data base (including CRP, Lrp, FIS, IHF, FNR, ArcA, NarL, GlpR, PurR, OmpR, TyrR, AraC, Fur, CytR, FruR,Hns, ArgR, DnaA, PhoB, and LexA), we could increase the sensitivity in an average of 35%, while retain almost the same sensitivity than a single position weight matrix (average correlation coefficient 87%).

RNA Polymerase Sites

(1) We gathered sigma 70 class I and class II promoters [20, 40] from the RegulonDB database and [17]. Then, we built models of the RNA polymerase site using a neuro-fuzzy method (see HPAM [19], [9]), and used the resulting models to perform genome-wide descriptions of the intergenic regions of the *E. coli* and *Salmonella* genomes with a false discovery rate <0.001 (see Promoter search in [19]). **(2)** We used an intelligent parser to differentiate class I and class II promoters that evaluate the quality of the -35 motif [20, 2], based on fuzzy logic and genetic algorithms techniques (see MOSS in [19]). **(3)** To characterize the distance relationship between transcription factors binding sites and RNA polymerase binding sites, we built models of such distances from the examples reported in the RegulonDB database. **(3.1)** We modeled activated and repressed promoters (see below *Activated or repressed* feature). **(3.2)** We re-built histograms for each group of distances (i.e. activated and repressed), distinguishing three overlapping distributions for each of them (Figure S3). **(3.3)** We built models for distances by fitting their distributions into models based on fuzzy membership functions [23] (see below), which were termed close, medium and remote distances for each set of activated and repressed genes. Finally, to characterize the distance relationship between the PhoP box and putative RNA polymerase binding site, we connected Steps (2) and (3) by using fuzzy logic-based operations (see below).

This process allowed us to retrieve the most representative RNA polymerase binding site candidates for each promoter region relative to the PhoP binding site (e.g., best class II RNA polymerase site, which is located close to the PhoP box in an activated promoter), which were arrayed and constituted the value of the RNA polymerase site feature in Figure S1. The probabilistic interpretation of the former process is usually the posterior probability (e.g., $p(class\,II/close)$ that, given a *close* promoter, it comes from class "class II" by following Bayes' rule [12, 6, 5]). This process is analogous to classification methods termed Naïve Bayes [33] if the T-norm and the T-conorm (see below) are restricted to the Product and the Maximum, respectively: $v_{MAP} = \arg\max_{v_j \in V} P(v_j) \prod_i P(a_i|v_j)$ where v_{MAP} (maximum a posteriori probability) denotes the target value output by the Naïve Bayes classifier and v_j, a_icorrespond to the features and attributes or variables, respectively.

Performance. The RNA polymerase site feature was evaluated using 721 RNA polymerase sites from RegulonDB as positive examples and 7210 random sequences as negative examples. We obtained an 82% sensitivity and 95% specificity for detecting RNA polymerase sites. These values provide an overall performance measurement (see below) of 92% corresponding to a false discovery rate <0.001 and a correlation coefficient of 82%. In addition, we selected 34 examples of RNA polymerase sites reported to be of class II, which all differ from the typical class I promoter by exhibiting a degenerate -35 sequence motif [32, 20, 2], and obtained 74% sensitivity and 95% specificity.

Activated/Repressed

We modeled PhoP-regulated promoters as activated or repressed based on examples reported in the RegulonDB database [39]. **(1)** We separately grouped activated and repressed promoters, and developed histograms for each group corresponding to the distances between transcription factor binding sites and the transcription initiation (+1) site. **(2)** We distinguished two non-disjoint distributions in each group and built models for these distances by fitting histograms with fuzzy membership functions [23] (Figure S4A-D), which do not force promoters to be exclusively Activated or Repressed. **(3)** Finally, we connected **(2)** and sigma 70 promoters previously detected to select the most representative candidate for each promoter condition (e.g., best promoter that characterize the activated condition) by using fuzzy logic-based operations as described above, which also have a probabilistic interpretation (e.g., $p(activated/sigma\,70)$), to characterize relationships between predicted PhoP and RNA polymerase binding sites detected in candidate promoters. Simple features, such as activated and repressed can be combined in more complex composite models to represent divergently transcribed genes (e.g., two adjacent genes, one repressed, the other activated, both sharing the same putative PhoP box in different orientations) using *fuzzy logic* expressions.

Binding Sites for other Transcription Factors

We developed models for different transcription factor binding sites from the RegulonDB database as follows: **(1)** We built position weight matrices for each transcription factor using the Consensus/Patser program, choosing the best final matrix for motif lengths between 14-30 bps if the corresponding length had not been previously specified (see "Consensus matrices" in [19]). We accounted for the motif symmetry (e.g., asymmetric, direct, inverted [40]) if available (see "Search known transcription factor motifs" in [19]). **(2)** We searched the intergenic regions of the *E. coli* and *Salmonella* genomes with these models, using the *overall performance* measure (see below) and additional 772 promoters from the RegulonDB database [39] to establish a threshold (average *E-value* $<$ 10E-10) for each matrix [4] (see "Thresholded consensus" in [19]). **(3)** We accounted for the distances between distinct transcription factors binding sites occurring in the same promoter region (e.g., the distance between the CRP and FIS sites in the *proP* promoter [31]) in promoters reported in RegulonDB database and built a histogram with the obtained results (Figure S4D). **(4)** We fitted the histogram using a fuzzy membership function (see below) and used this model as a fuzzy cluster to characterize the distances between a putative PhoP box and another putative transcription factor binding site detected in the same region. **(5)** Finally, we connected **(2)** and **(4)** by using fuzzy logic-based operations as described above, which can also have a probabilistic interpretation (e.g., *p(CRP,FIS/appropriate distance*) upstream of the *proP* open reading frame of *E. coli*), to characterize PhoP regulated candidates promoters.

Fuzzy Logic Expressions

Propositional calculus logic expressions [37] can be extended by incorporating predicates having fuzzy variables, which are manipulated using various theorems/axioms and methods [23]. This approach, which has been widely used in several fields including decision-making [7], artificial intelligence [52] and electrical engineering [10] for many years, was applied to model related features that describe different regulatory objects.

Thus, given a dataset $X = \{x_1, ..., x_n\}$, the feature that characterizes it can be best described as a set $F_1(X) = \{d_{11}/x_1, ..., d_{1n}/x_n\}$, where $\{d_{11}, ..., d_{1n}\} \in \{0, 1\}$ in classical set theory and [0,1] in fuzzy set theory. These fuzzy values represent the degree of matching between an observation of the dataset and a fuzzy set. The degree of matching is defined in the unit interval and can be obtained from evaluating the membership function of the corresponding fuzzy set (see below). Then, given $F_2(X) = \{d_{21}/x_1, ..., d_{2n}/x_n\}$ and the Minimum as an intersection operator, we define the expression:

$$\begin{aligned} F_1(X) \bigcap F_2(X) &= MIN(F_1, F_2) \\ &= \{MIN(d_{11}, d_{21})/x_1, ..., MIN(d_{1n}, d_{2n})/x_n\}. \end{aligned}$$

Fuzzy logic-based operations, such as T-norms/conorms, include operators like *MINIMUM, PRODUCT*, or *MAXIMUM*, which are used as basic logic operators, such as AND or OR, or their set equivalents *INTERSECTION* or *UNION* [5, 23]. We used in this work the Minimum and Maximum as T- and T-conorms, respectively.

Fuzzy Membership Functions

They can be viewed as approximation of data distributions, where the degree of matching in the [0,1] scale is calculated using triangular functions [23]:

$$\mu(x) = \begin{cases} 0, & \text{if } x < a_0 \text{ or } x > a_2 \\ (x - a_0)/(a_1 - a_0), & \text{if } x < a_1 \\ (a_2 - x)/(a_2 - a_1), & \text{if } x > a_1 \\ 1, & \text{otherwise} \end{cases} \tag{3.1}$$

where $a_0, ..., a_2$ are learned from the projection of the histograms onto the variable domains (Figure S4) by simple regression and minimum squared methods [12, 47]. This process is analogous to fitting histograms to a distribution, and assigning probability values based on a density function. Our approach, however, adopts a distribution-independent and non-parametric fitting process by projecting data [12, 47] into triangular functions.

Fuzzy C-Means Clustering Method

Fuzzy C-Means clustering method [6, 13] is an extension of the K-means clustering method, where the elements can belong to more than one cluster with a

different degree of membership. Thus, membership of a promoter $k$with a feature valuex_k in a particular cluster V_i is calculated as:

$$\mu_{i,k} = \left[1 + \left(\|x_k - V_i\|_A^2 \Big/ w_i \right)^{1/m-1}\right]^{-1} \quad \forall i,k;\ 1 < m \leq \infty \tag{3.2}$$

where c-partitions of the data X were usually arrayed as a (cxn) matrix Ucontaining the vector representation of matching between n promoters and c-partitions of one type of feature, $\mu_{i,k}$ is taken as the degree of membership of the value x_k in the ith partitioning fuzzy subset of X, $\overline{V_i}$is the cluster prototype or centroid of partition V_i, $m$is the degree of fuzzification, A determines the type of norm commonly used in pattern recognition (e.g., A=1 is the city block norm; A=2 is the Euclidean norm, etc [5]), and a weight for penalty terms w_i, which is initialized as 1 in the absence of external information. If the approach is probabilistic $\mu_{i,k}$ is usually the posterior probability $p(i/x_k)$ that, given x_k, it comes from class i by following Bayes' rule [6, 5, 33]. If the approach is fuzzy, x_k can come from more than one class, and if it is possibilistic [5], a more realistic situation is represented where we do not force each element to belong to a class. The cluster prototype or centroid of partition V_i is calculated as:

$$V_i = \frac{\sum_{k=1}^{n} (\mu_{i,k})^m x_k}{\sum_{k=1}^{n} (\mu_{i,k})^m} \quad \forall i \tag{3.3}$$

based on the use of the Euclidean distance as a similarity function:

$$\|x, V\|_2 = \sqrt{(x-V)^T (x-V)} \tag{3.4}$$

Summarizing, (0) Initialize $V_0 = \{v_1, ..., v_c\}$ (1) While (t < T and $\|V_t - V_{t-1}\| > \varepsilon$) (2) Calculate U_t with V_{t-1}, (3) update V_{t-1} to V_t, with U_t (4) Iterate, where T is the maximum number of iterations.

Xie-Beni Validity Index [5]

The minimization of this index through different number of clusters (i.e., $c = 2$to $c = \sqrt{n}$) detects compact representations of Fuzzy C-Means partitions:

$$XB(U,L) = \frac{\sum_{k=1}^{n} \sum_{i=1}^{c} u_{i,k}^2 \left\|x_k - \overline{V_i}\right\|^2}{n \left(\min_{i \neq j} \left\{ \left\|\overline{V_i} - \overline{V_j}\right\|^2 \right\} \right)}. \tag{3.5}$$

Performance Measurement

We use a correlation coefficient implementation to establish best local thresholds for transcription factor binding site motifs. That is, from a range of possible thresholds applied over a particular motif, we choose the one that maximizes this coefficient defined as:

$$CC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP) \times (TN + FN) \times (TP + FN) \times (TN + FP)}}, \quad (3.6)$$

where *specificity = TN/(TN + FP)* and *sensitivity = TP/(TP + FN)*; *P= positive, N=negative, T = true and F=false* [4]. We constrained the sensitivity of the selected threshold to be above the 60%. The false positive rate for binding site analysis was calculated by detecting binding sites from other transcription factors different from the one being evaluated (RegulonDB database).

Dataset

We initially used the intergenic regions of *E. coli* and *Salmonella* operons from -800 to +50 because >5% are larger than 800 bp in bacterial genomes (as described in the RegulonDB database or generously provided by H. Salgado) [40]; however, predictions have been performed in whole coding and non coding regions (see [19]). The promoter and transcription factor information was taken from RegulonDB database. We compiled from the literature and our own lab information (Table S1) genes whose expression (using microarrays) differed statistically between wild-type and *phoP E. coli* strains experiencing inducing conditions for the PhoP/PhoQ regulatory system [53], as well as a list of genes known/assumed to be PhoP regulated (Table S2). However, this information did not explicitly indicate whether these genes were regulated directly or indirectly by the PhoP protein. The learned features were used to make genome-wide predictions in the *E. coli* and *Salmonella* genomes.

Programming Resources

The scripts and programs used in this work, some of which are accessible from [19], were based on Perl, Matlab r2006a and C++ interpreters/languages, and the visualization routines were performed on Spotfire DecisionSite software 8.2. Data and predictions for *E. coli* and *Salmonella* genomes will be available at [19].

Bacterial Strains, Plasmids and Growth Conditions

Bacterial strains and plasmids used in this study are listed in Table S3. *Salmonella enterica* serovar Typhimurium strains used in this study are derived from strain 14028s. Bacteria were grown at 37 $^\circ$ C in Luria-Bertani broth (LB) [41] or N-minimal medium pH 7.7 [44] supplemented with 0.1% Casamino Acids, 38 mM glycerol, $MgCl_2$. Kanamycin was used at 25 μg/ml.

Constructions of GFP Reporter Plasmids

Promoter regions (i.e. the intergenic region between two ORFs) were amplified using PCR. A list of the promoter-specific primers used in the PCR reactions is shown in Table S4. The PCR fragment was digested with *Bam*HI and *Xho*I, purified, then introduced to the cloning site of pMS201 (GFP reporter vector plasmid, a gift from Alon, U [28]). Sequences of promoter region were verified by nucleotide sequencing.

Measurements of Promoter Activity and Growth Kinetics for GFP Reporter Strains with High-Temporal Resolution

Promoter activity and growth kinetics of wild-type *Salmonella* strain harboring GFP reporter plasmid was measured in parallel using automated microplate reader (VICTOR^3, Perkin Elmer) [28]. Overnight cultures of strains in N-minimal medium with 10 mM $MgCl_2$ and 25 μg/ml of kanamycin were washed with the same medium without $MgCl_2$ then diluted (1:100) to 96-well plate (Packard) containing 150 μl of N-minimal media supplemented 50 μM $MgCl_2$. After overlaying the wells with 50 μl of mineral oil (Sigma) to prevent evaporation of media, the plate was inserted in the VICTOR^3 machine pre-warmed to 37 $^\circ$C. The fluorescence and optical density (600 nm) of cells were recorded with shaking of the plate (1 min with 0.1 mm diameter), and this protocol was repeated every 6 min for 99 times. The background fluorescence was measured using a strain carrying empty vector and subtracted from the test values. Each experiment was conducted independently twice (Figure S6), and a representative is shown in the figures.

Data Preprocessing

The raw GFP and OD signals were used to calculate the promoter activity as $[dG_i(t)/dt]/OD_i(t)$. The activity signal was then smoothed by a shape-preserving interpolant (Piecewise Cubic Hermite Interpolating Polynomial, Matlab r2006a) fitting algorithm that finds values of an underlying interpolating function at intermediate points that are not described in the experimental assays. Then, we applied a polynomial fit (sixth order, Matlab r2006a) on each expression signal. This smoothing procedure captures the dynamics well, while removing the noise inherent in the differentiation of noisy signals.

3.3 Results and Discussion

Approach

We investigated five types of *cis*-acting promoter features by extracting the maximal amount of useful information from datasets and then creating models that describe promoter regulatory regions. This entailed applying three key strategies: first, we conducted an initial survey of the data provided from different available sources, capturing and distinguishing between broad and easily discernable patterns. We then used these patterns as models to re-visit the data with greater sensitivity and specificity, which allowed the detection of those instances where a binding site sequence resembles the consensus only weakly or where the distances between the transcription factor and the RNA polymerase are unusual. Second, we utilized fuzzy clustering methods [5, 13] to encode how a promoter matches each of the multiple models for a given promoter feature, which avoided having to make premature categorical assignments, thus producing an initial classification of the promoters into multiple subsets. Finally, we applied fuzzy logic [23] to link basic features into more informative composite models that explain the distinct expression behavior of genes belonging to similar networks (Figure S1).

Transcription Factor Binding Site Submotifs

Many genes are controlled by a single-input network motif where the affinity of a transcription factor for its promoter sequences is a major determinant of gene expression (Figure 3.2(A)). Thus, co-regulated genes displaying distinct expression patterns are likely to differ in the binding site for such a transcription factor. Methods that look for matching to a consensus sequence have been successfully used to identify promoters controlled by particular transcription factors [1, 46, 29]. However, the strict cutoffs used by such methods increase specificity but decrease sensitivity [18, 46], which makes it difficult to detect binding sites with weak resemblance to a consensus sequence.

To circumvent the limitation of consensus methods [49], we decomposed the binding site motif of a transcription factor into several submotifs and then combined the submotifs into a multi-classifier (see Methods), which increased the sensitivity to weak sites without losing specificity. In the case of PhoP, we identified four submotifs (Figure S2), and used them to search both strands of the intergenic regions of the *E. coli* and *Salmonella* genomes (Figure S1). This allowed the recovery of promoters, such as that corresponding to the *E. coli hdeA* gene or the *Salmonella pmrD*, that had not been detected by the single consensus position weight matrix model [18, 46] despite being footprinted by the PhoP protein [53, 11, 32], [22, 34, 24].

To test the notion that PhoP binding to promoters with different PhoP box submotifs is a determinant promoter activity, we compared the gene expression patterns of wild-type *Salmonella* harboring plasmids with a transcriptional fusion between a promoterless *gfp* gene to different PhoP-activated promoters. Faster GFP expression kinetics were observed when transcription was driven by the *phoP* promoter, which has the M2 submotif, than when it was driven by the *pmrD* promoter, which has the M1 submotif, (Figure 3.2 (B)-(C)). Thus, the binding site for a transcriptional regulator is a key determinant in gene expression.

The use of submotifs instead of a single consensus increased the sensitivity for PhoP binding sites from 66% to 91%; yet, the specificity remained essentially the same (i.e., 98% in a consensus model versus 97% in the case of submotifs). Importantly, this approach is not exclusive to binding sites recognized by the PhoP protein as the sensitivity for sites recognized by the cAMP receptor protein (CRP) increased from 29% to 50% when submotifs were used instead of a consensus; yet, the specificity remained the same at 98%.

Transcription Factor Binding Site Orientation

Functional binding sites for a transcription factor may be present in either orientation relative to the RNA polymerase binding site [43]. This is due to the possibility of DNA looping and to the flexibility of the alpha subunit of the bacterial RNA polymerase in its interactions with transcriptional regulators [2, 48].

Analysis of PhoP-regulated promoters revealed that the PhoP box could be found with the same probability in either orientation in the intergenic regions of the *E. coli* and *Salmonella* genomes (Figure S5). For example, the *E. coli*

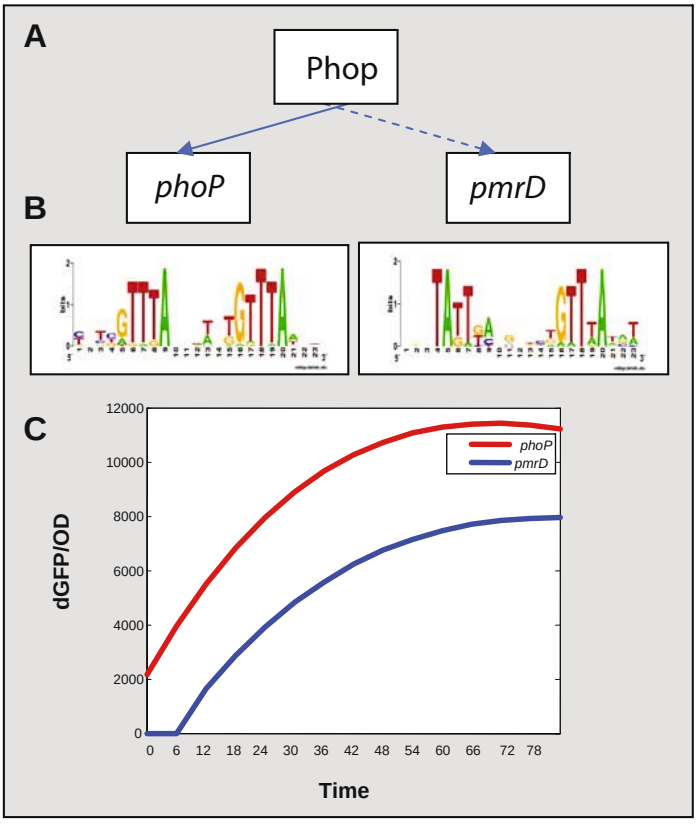


Fig. 3.2. The PhoP protein achieves differential expression using the single-input network motif by controlling genes that differ in their binding site submotifs. (a) PhoP regulates several promoters (i.e. *phoP* and *pmrD*) using a single-input network motif. (b) The PhoP protein recognizes a binding site motif consisting of a hexameric direct repeat separated by 5 bp, but distinguishes between different submotifs with different specificities. We identified four of these classes (M_1- M_4; Figure S2), and tested the influence of this *cis*-feature in the *phoP* and *pmrD Salmonella* promoters corresponding to class M_2 and M_1, respectively. (c) Transcriptional activity of wild-type *Salmonella* harboring plasmids with a transcriptional fusion between a promoterless *gfp* gene and the *Salmonella phoP* (red color) or *pmrD* (blue color) promoters. The activity of each promoteris proportional to the number of GFP molecules producedper unit time per cell $[dG_i(t)/dt/\mathrm{OD}_i(t)]$, where$G_i(t)$ is GFP fluorescence from wild-type *Salmonella* strain 14028s culture and conditions described in Methods, and $OD_i(t)$ is the optical density. The activity signal was smoothed by a polynomial fit (sixth order). The results are not normalized. Faster and earlier GFP expression was observed when transcription was driven by the *phoP* promoter, which has the M2 submotif, than by the *pmrD* promoter, which has the M1 submotif.

ompT and *yhiW* promoters and the *Salmonella mig-14, pipD*, *pagC* and *pagK* promoters harbor putative PhoP binding sites in the opposite relative orientation to that described for the prototypical PhoP-activated *mgtA* promoter [53] (Figure S1). Yet other promoters (i.e. those of the *ybjX*, *slyB*, *yeaF* genes in *E. coli* and the *virK*, *ybjX,* and *mgtC* genes in *Salmonella*) contain sequences resembling the PhoP box in both orientations. The demonstration that PhoP does bind to the *mgtC, mig-14* and *pagC* promoters [53], which harbor the PhoP binding site in the opposite orientation as in the *mgtA* promoter, validates our predictions and argues against alternative network designs where these promoters would be regulated by PhoP only indirectly [25].

To assess the contribution of PhoP box orientation to gene expression, we determined the fluorescence of wild-type *Salmonella* harboring plasmids with a transcriptional fusion between a promoterless *gfp* gene to PhoP-regulated promoters that differed in the orientation of the PhoP box. Promoters with the PhoP box in the direct orientation, such as those corresponding to the *yobG* and *slyB* genes, were transcribed earlier and faster than the *pagK* and *pagC* promoters in which the PhoP box is in the opposite relative orientation (Figure 3.3(A)-(C)). This is in spite of the fact that *yobG* and *pagK* promoters are equally divergent from the PhoP binding site consensus (60% and 66% of the consensus information content (Figure S2), respectively). Furthermore, promoters sharing the same PhoP binding site submotif but arranged in different orientations (e.g. the *ugd* and *mig-14* promoters) produced distinct rise times and expression levels (data not shown).

RNA Polymerase Site

The distance of a transcription factor binding site to the RNA polymerase binding site(s) and the class of sigma 70 promoter are critical determinants of gene expression [2]. These classes correspond to the different types of contacts that can be established between a transcription factor and RNA polymerase.

We identified seven patterns among PhoP-regulated promoters of *E. coli* and *Salmonella* (Figure S1) that combine promoter class and distance between the PhoP box and the RNA polymerase site (Figure S3). These patterns may correspond to different kinetic behaviors within a network motif [2]. For example, the *ugtL* and *pagC* promoters share the orientation of the PhoP box but differ in the distance of the PhoP box to the RNA polymerase binding site (Figure 3.4(A)-(B)). This may account for the different dynamic behavior of these promoters when tested in a wild-type strain harboring plasmids with promoter fusions to the promoterless *gfp* gene (Figure 3.4(C)).

In addition, some PhoP-regulated promoters (e.g. the *hemL* and *phoP* promoters of *E. coli*) contain several putative RNA polymerase binding sites located at different positions and belonging to different classes, suggesting that such promoters may be regulated by additional signals and/or transcription factors [32].

Activated/Repressed Promoters

Gene expression data normally allow clear separation of genes into those that are activated and those that are repressed by a regulatory protein. Because the

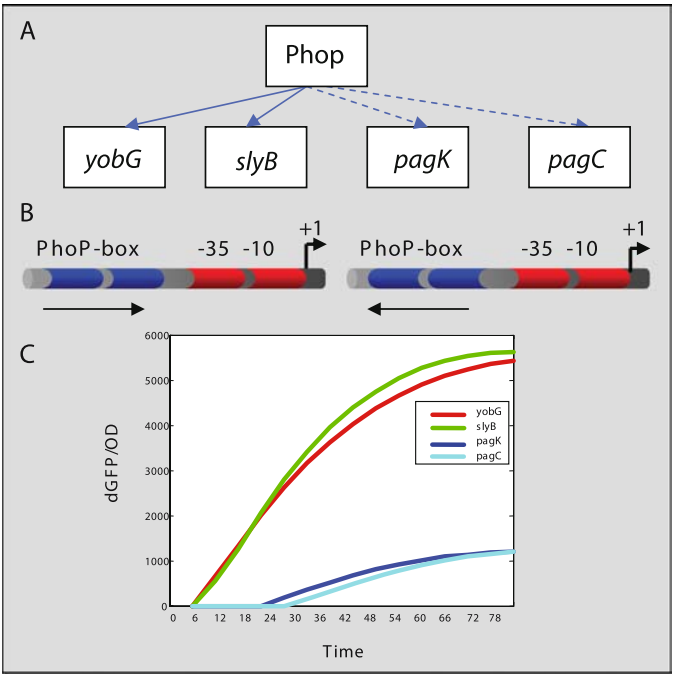


Fig. 3.3. Expression of PhoP-regulated promoters that differ in the orientation of the PhoP-binding site. (a) PhoP regulates a set of promoters including those of the *Salmonella yobG*, *slyB*, *pagK* and *pagC* genes using a single-input network motif. (b) We established that when *Salmonella* experiences low Mg^{2+}, the PhoP protein binds to both the archetypal directly oriented *yobG* and *slyB* promoters as well as the oppositely oriented *pagK* and *pagC* promoters using chromatin immunoprecipitation (ChIP) *in vivo*. (c) Transcriptional activity of wild-type *Salmonella* harboring plasmids with a transcriptional fusion between a promoterless *gfp* gene and the *Salmonella yobG* (red color) or *slyB* (green color) promoters reveals a much earlier an higher levels of activity than the isogenic strains with fusions to the *pagK* (blue color) and *pagC* (cyan color) promoters. Promoter activity was determined as described in the legend to Figure 3.2. Thus, the orientation of the binding site for a transcriptional regulator contributes to the kinetic behavior as well as the maximum expression levels achieved by the promoters.

expression signal is sometimes absent or too low to be informative, we considered the location of a transcription factor binding site relative to that of the RNA polymerase to separate promoters into activated and repressed subsets (Figure S4A-C) [8].

We determined that the location of binding sites functioning in activation is different from that corresponding to sites functioning in repression (Figure S4A-C), being centered ∼40 and ∼20 bp upstream of the transcription start site, respectively. This allowed us to distinguish among PhoP-regulated promoters that have apparently similar network motifs (Figure S1). For ex-

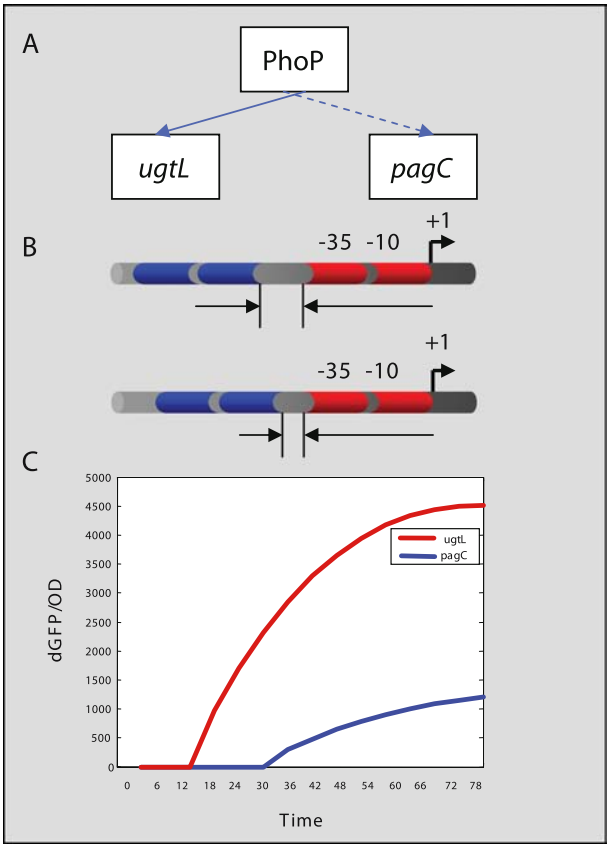


Fig. 3.4. Expression of PhoP-regulated promoters that differ in the RNA polymerase sites. (a-b) The PhoP-activated *ugtL* and *pagC* promoters share the orientation of the PhoP-binding site as well as the class I sigma 70 promoter, but differ in the distance between the PhoP box and the RNA polymerase site. (c) Transcriptional activity of wild-type *Salmonella* harboring plasmids with a transcriptional fusion between a promoterless *gfp* gene and the *Salmonella ugtL* (red color) and *pagC* (blue color) promoters. Promoter activity was determined as described in the legend to Figure 3.2. The *ugtL* promoter is transcribed earlier than the *pagC* promoter, which also exhibits a PhoP box in the opposite orientation but more distant from the RNA polymerase site.

ample, we identified a PhoP binding site at a relative distance to the RNA polymerase consistent with repression in the promoter region of the *hilA* gene, which encodes a master regulator of *Salmonella* invasion and had been known to be under transcriptional repression by the PhoP/PhoQ system [15, 42]. Several promoters, including those of the *Salmonella pipD* and *nmpC* genes, were classified as candidates for being both activated and repressed, because the distance between the predicted transcription start site and the PhoP box is consistent with either activation or repression. Gene expression experiments conducted in

E. coli indicate that *nmpC* is a PhoP-repressed gene [53, 11, 32]. Other promoters were predicted to have more than one PhoP box (e.g., those of the PhoP-activated *mgtC* and *pagC* genes), where one could correspond to an activation site and the other to a repression site. Indeed, this appears to be the case of the PhoP-activated *iraP* gene [50].

Binding Sites for other Transcription Factors

Certain promoters harbor binding sites for more than one transcription factor. This could be because transcription requires the concerted action of such proteins, or because the promoter is independently activated by individual transcription factors, each responding to a distinct signal.

We analyzed the intergenic regions of the *E. coli* and *Salmonella* genomes for the presence of binding sites for 54 transcription factors [39]. We then investigated the co-occurrence of 24 sites with the binding site of the PhoP protein in an effort to uncover different types of network motifs involving PhoP-regulated promoters. For example, the *Salmonella pmrD*, *ugd* and *yrbL* promoters and the *E. coli yrbL* promoter harbor PhoP- and PmrA-binding sites, consistent with the experimentally-verified regulation by both the PhoP and PmrA proteins that can be described by the bi-fan network motif [53, 21] (Figure 3.5(A)). In addition, the relative position of transcription factor binding sites (Figure S4D) can play a critical role because the PmrA-box in the *Salmonella pmrD* and *yrbL* promoters is located closer to the PhoP-box ($\sim$38 bp and $\sim$24 bp, respectively) than in the *udg* promoter ($\sim$65 bp), which could account for the different expression patterns exhibited by their respective genes (Figure 3.5(B)-(C)). By analyzing both the binding site quality and the location of transcription factor binding sites, we increase the chances of identifying co-regulated promoters.

By considering the presence of binding sites for multiple transcription factors, it is possible to generate hypotheses about potential network motifs. For example, the promoters of the PhoP-activated *gadA*, *dps*, *hdeA*, *yhiE* and *yhiW* genes of *E. coli* also have binding sites for the regulatory proteins YhiX and YhiE [53], raising the possibility that some of these genes might be regulated by feedforward loops where both the PhoP protein and either the YhiW or the YhiE proteins would bind to the same promoter to activate transcription. This notion was experimentally verified [53], validating our prediction.

Regulation of Orthologous Genes

A distinguishing characteristic of our approach is that promoters for orthologous genes are considered individually. This is in contrast to some phylogenetic footprinting methods [30] that often ignore regulatory differences among closely-related organisms due to their strict reliance on the conservation of regulatory motifs across bacterial species. Thus, we could uncover cases of phenotypic differences between closely-related species resulting from the differential regulation of homologous genes. For example, the *ugd* and *iraP* promoters of *Salmonella* (Figure S1) harbor functional PhoP boxes that are footprinted by the purified PhoP protein [22, 34, 24] [50]. By contrast, the PhoP boxes are missing from

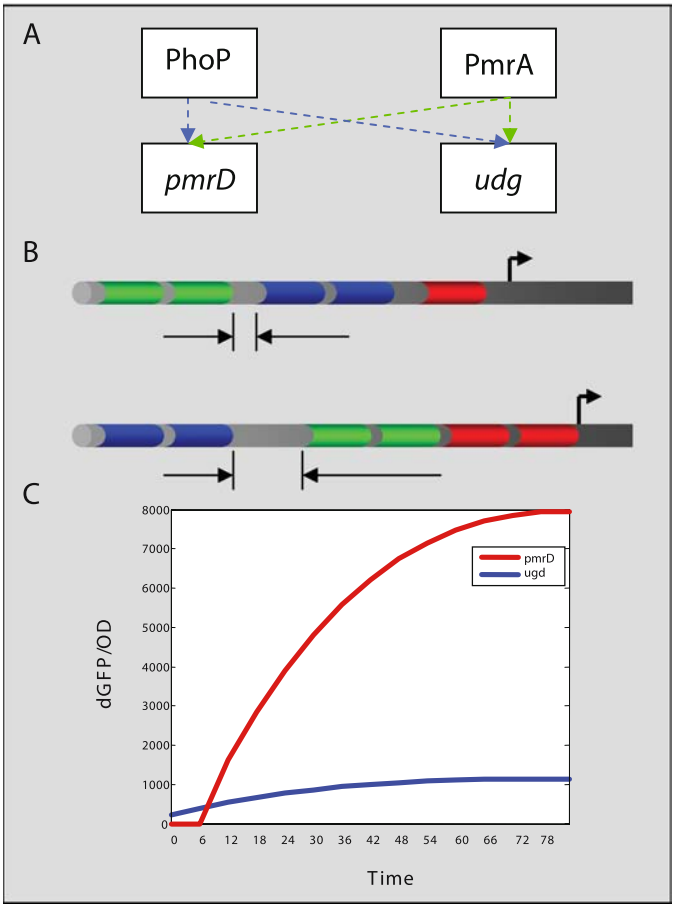


Fig. 3.5. Expression of PhoP-regulated promoters that use the bi-fan network motif. (**a**) The *Salmonella pmrD*, and *ugd* promoters harbor experimentally verified PhoP- and PmrA-binding sites that can be described by the bi-fan network motif. (**b**) The distance between the PhoP and PmrA boxes in the *Salmonella pmrD* and *ugd* promoters are different (∼38 bp and ∼65 bp, respectively). (**c**) Transcriptional activity of wild-type *Salmonella* harboring plasmids with a transcriptional fusion between a promoterless *gfp* gene and the *Salmonella pmrD* and *ugd* promoters. Promoter activity was determined as described in the legend to Figure 3.2. The two promoters confer different expression and kinetic patterns.

the *ugd* and *iraP* promoters of *E. coli*, preventing it from expressing these genes under the same conditions as *Salmonella, Tu06* (Mouslim and Groisman, unpublished results). Likewise, there is a PhoP box in the *pmrD* promoters of both *E. coli* and *Salmonella* (albeit of different submotifs) but only the *Salmonella pmrD* promoter has a PmrA box that functions as a repression site [51, 53]. This

demonstrates that the detailed analysis of *cis*-features can shed light on different network motif design among closely-related bacterial species.

3.4 Conclusions

We demonstrated that a transcription factor can mediate differential expression of genes that are described by the same network motif. This is because of the functional significance of variability in sequence, location and topology that exists among promoters that are co-regulated by a given transcription factor. We developed a flexible computational framework to encode and to combine these promoter features, which allows matching of *cis*-observations to multiple models for a given promoter feature. This enables the description of regulatory elements from different angles and the generation of composite models that can be used to explain the different kinetic behavior of co-regulated genes.

Finally, unlike regulators such as the LacI [29] and MelR [14] proteins of *E. coli* that govern expression of single promoters, many transcriptional regulators control multiple promoters that express products required in different amounts or for different extents of time. This is clearly the case for the regulatory protein PhoP, which controls transcription of a large numbers of genes, that can be described by a variety of network motifs (Figure 3.1). Our findings argues that understanding a cell's behavior in terms of differential expression of genes controlled by a transcription factor requires a detailed analysis of a promoter's regulatory features. As a single nucleotide difference in the binding site for a transcription factor can dictate the requirement for co-activator proteins [26], we feel that by considering multiple models (as opposed to the relying on consensuses) it will be possible to uncover subtle differences between regulatory targets and to capture the salient properties of co-regulated promoters.

Acknowledgments

This work was partly supported by the Spanish Ministry of Science and Technology under Project BIO2004-0270-E, and I.Z. is also supported by and by Howard Hughes Medical Institute. O.H. acknowledges the doctoral MAEC- AECI fellowship.

References

1. Bailey, T.L., Elkan, C.: The value of prior knowledge in discovering motifs with MEME. In: Proc. Int. Conf. Intell. Syst. Mol. Biol., vol. 3, pp. 21–29 (1995)
2. Barnard, A., Wolfe, A., Busby, S.: Regulation at complex bacterial promoters: how bacteria use different promoter organizations to produce different regulatory outcomes. Curr. Opin. Microbiol. 7, 102–108 (2004)
3. Beer, M.A., Tavazoie, S.: Predicting gene expression from sequence. Cell 117, 185–198 (2004)

4. Benitez-Bellon, E., Moreno-Hagelsieb, G., Collado-Vides, J.: Evaluation of thresholds for the detection of binding sites for regulatory proteins in Escherichia coli K12 DNA. Genome. Biol. 3, RESEARCH0013 (2002)
5. Bezdek, J.C.: Pattern analysis. In: Ruspini, E.H., Pedrycz, W., Bonissone, P.P. (eds.) Handbook of Fuzzy Computation, pp. F6.1.1–F6.6.20. Institute of Physics, Bristol (1998)
6. Bezdek, J.C., Pal, S.K.: Fuzzy models for pattern recognition: methods that search for structures in data. IEEE Press, New York (1992)
7. Cipra, B.: Mathematicians offer answers to everyday conundrums. Science 283, 927–935 (1999)
8. Collado-Vides, J., Magasanik, B., Gralla, J.D.: Control site location and transcriptional regulation in Escherichia coli. Microbiol. Rev. 55, 371–394 (1991)
9. Cotik, V., Zaliz, R.R., Zwir, I.: A hybrid promoter analysis methodology for prokaryotic genomes. Fuzzy Sets and Systems 152, 83–102 (2005)
10. Cuesta, F., Gomez-Bravo, A., Ollero, A.: Parking maneuvers of industrial-like electrical vehicles with and without trailer. IEEE Transactions on Industrial Electronics 51, 257–269 (2004)
11. Eguchi, Y., Okada, T., Minagawa, S., Oshima, T., Mori, H., Yamamoto, K., Ishihama, A., Utsumi, R.: Signal Transduction Cascade between EvgA/EvgS and PhoP/PhoQ Two-Component Systems of Escherichia coli. J. Bacteriol. 186, 3006–3014 (2004)
12. Everitt, B., Der, G.: A handbook of statistical analysis using SAS. Chapman & Hall, London (1996)
13. Gasch, A.P., Eisen, M.B.: Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. Genome. Biol. 3, RESEARCH0059 (2002)
14. Grainger, D.C., Overton, T.W., Reppas, N., Wade, J.T., Tamai, E., Hobman, J.L., Constantinidou, C., Struhl, K., Church, G., Busby, S.J.: Genomic studies with Escherichia coli MelR protein: applications of chromatin immunoprecipitation and microarrays. J. Bacteriol. 186, 6938–6943 (2004)
15. Groisman, E.A.: The pleiotropic two-component regulatory system PhoP-PhoQ. J. Bacteriol. 183, 1835–1842 (2001)
16. Groisman, E.A., Heffron, F., Solomon, F.: Molecular genetic analysis of the Escherichia coli phoP locus. J. Bacteriol. 174, 486–491 (1992)
17. Harley, C.B., Reynolds, R.P.: Analysis of E. coli promoter sequences. Nucleic Acids Res. 15, 2343–2361 (1987)
18. Hertz, G.Z., Stormo, G.D.: Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. Bioinformatics 15, 563–577 (1999)
19. `http://gps-tools2.wustl.edu`
20. Ishihama, A.: Protein-protein communication within the transcription apparatus. J. Bacteriol. 175, 2483–2489 (1993)
21. Kato, A., Groisman, E.A.: Connecting two-component regulatory systems by a protein that protects a response regulator from dephosphorylation by its cognate sensor. Genes. Dev. 18, 2302–2313 (2004)
22. Kato, A., Latifi, T., Groisman, E.A.: Closing the loop: the PmrA/PmrB two-component system negatively controls expression of its posttranscriptional activator PmrD. Proc. Natl. Acad. Sci. USA 100, 4706–4711 (2003)
23. Klir, G.J., Folger, T.A.: Fuzzy sets, uncertainty, and information. Prentice Hall International, London (1988)

24. Latifi, Y.S.N.S.T., Cromie, M.J., Groisman, E.A.: Transcriptional control of the antimicrobial peptide resistance ugtL gene by the Salmonella PhoP and SlyA regulatory proteins. J. Biol. Chem. 279, 38618–38625 (2004)
25. Lejona, S., Aguirre, A., Cabeza, M.L., Garcia Vescovi, E., Soncini, F.C.: Molecular characterization of the Mg2+-responsive PhoP-PhoQ regulon in Salmonella enterica. J. Bacteriol. 185, 6287–6294 (2003)
26. Leung, T.H., Hoffmann, A., Baltimore, D.: One nucleotide in a kappaB site can determine cofactor specificity for NF-kappaB dimers. Cell 118, 453–464 (2004)
27. Pritsker, M., Liu, Y.C., Beer, M.A., Tavazoie, S.: Whole-genome discovery of transcription factor binding sites by network-level conservation. Genome Research, 99–108 (2004)
28. Mangan, S., Alon, U.: Structure and function of the feed-forward loop network motif. Proc. Natl. Acad. Sci. USA 100, 11980–11985 (2003)
29. Martinez-Antonio, A., Collado-Vides, J.: dentifying global regulators in transcriptional regulatory networks in bacteria. Curr. Opin. Microbiol. 6, 482–489 (2003)
30. McCue, L., Thompson, W., Carmack, C., Ryan, M.P., Liu, J.S., Derbyshire, V., Lawrence, C.E.: Phylogenetic footprinting of transcription factor binding sites in proteobacterial genomes. Nucleic Acids Res. 29, 774–782 (2001)
31. McLeod, S.M., Aiyar, S.E., Gourse, R.L., Johnson, R.C.: The C-terminal domains of the RNA polymerase alpha subunits: contact site with Fis and localization during co-activation with CRP at the Escherichia coli proP P2 promoter. J. Mol. Biol. 12, 517–529 (2002)
32. Minagawa, S., Ogasawara, H., Kato, A., Yamamoto, K., Eguchi, Y., Oshima, T., Mori, H., Ishihama, A., Utsumi, R.: Identification and molecular characterization of the Mg2+ stimulon of Escherichia coli. J. Bacteriol. 185, 3696–3702 (2003)
33. Mitchell, T.M.: Machine learning. McGraw-Hill, New York (1997)
34. Mouslim, C., Latifi, T., Groisman, E.A.: Signal-dependent requirement for the co-activator protein RcsA in transcription of the RcsB-regulated ugd gene. J. Biol. Chem. 278, 50588–50595 (2003)
35. Perron-Savard, P., De Crescenzo, G., Le Moual, H.: Dimerization and DNA binding of the Salmonella enterica PhoP response regulator are phosphorylation independent. Microbiology 151, 3979–3987 (2005)
36. Robison, K., McGuire, A.M., Church, G.M.: A comprehensive library of DNA-binding site matrices for 55 proteins applied to the complete Escherichia coli K-12 genome. J. Mol. Biol. 284, 241–254 (1998)
37. Rosenblueth, D.A., Thieffry, D., Huerta, A.M., Salgado, H., Collado-Vides, J.: Syntactic recognition of regulatory regions in Escherichia coli. Comput. Appl. Biosci. 12, 415–422 (1996)
38. Ruspini, E.H., Zwir, I.: Automated generation of qualitative representations of complex objects by hybrid soft-computing methods. In: Pal, S.K., Pal, A. (eds.) Pattern recognition: from classical to modern approaches, pp. 454–474. World Scientific, New Jersey (2002)
39. Salgado, H., Gama-Castro, S., Martinez-Antonio, A., Diaz-Peredo, E., Sanchez-Solano, F., Peralta-Gil, M., Garcia-Alonso, D., Jimenez-Jacinto, V., Santos-Zavaleta, A., Bonavides-Martinez, C., Collado-Vides, J.: RegulonDB (version 4.0): transcriptional regulation, operon organization and growth conditions in Escherichia coli K-12. Nucleic Acids Res. 32, D303–D306 (2004)
40. Salgado, H., Santos-Zavaleta, A., Gama-Castro, S., Millan-Zarate, D., Diaz-Peredo, E., Sanchez-Solano, F., Perez-Rueda, E., Bonavides-Martinez, C., Collado-Vides, J.: RegulonDB (version 3.2): transcriptional regulation and operon organization in Escherichia coli K-12. Nucleic Acids Res. 29, 72–74 (2001)

41. Sambrook, J., Fritsch, E.F., Maniatis, T.: Molecular cloning: a laboratory manual. Cold Spring Harbor Laboratory Press, New York (1989)
42. Schechter, L.M., Damrauer, S.M., Lee, C.A.: Two AraC/XylS family members can independently counteract the effect of repressing sequences upstream of the hilA promoter. Mol. Microbiol. 32, 629–642 (1999)
43. Lobell, R.B., Schleif, R.F.: AraC-DNA looping: orientation and distance-dependent loop breaking by the cyclic AMP receptor protein. J. Mol. Biol. 218, 45–54 (1991)
44. Snavely, M.D., Gravina, S.A., Cheung, T.T., Miller, C.G., Maguire, M.E.: Magnesium transport in salmonella typhimurium. Regulation of mgtA and mgtB expression. J. Biol. Chem. 266, 824–829 (1991)
45. Soncini, F.C., Garcia Vescovi, E., Solomon, F., Groisman, E.A.: Molecular basis of the magnesium deprivation response in Salmonella typhimurium: identification of PhoP-regulated genes. J. Bacteriol. 178, 5092–5099 (1996)
46. Stormo, G.D.: DNA binding sites: representation and discovery. Bioinformatics 16, 16–23 (2000)
47. Sugeno, M., Yasukama, T.: A fuzzy-logic-based approach to qualitative modeling. IEEE Transactions on Fuzzy Systems 1, 7–31 (1993)
48. Teichmann, S.A., Babu, M.M.: Conservation of gene co-regulation in prokaryotes and eukaryotes. Trends Biotechnol. 20, 407–410 (2002)
49. Tompa, M., Li, N., Bailey, T.L., Church, G.M., De Moor, B., Eskin, E., Favorov, A.V., Frith, M.C., Fu, Y., Kent, W.J.W.J., Makeev, V.J., Mironov, A.A., Noble, W.S., Pavesi, G., Pesole, G., Regnier, M., Simonis, N., Sinha, S., Thijs, G., van Helden, J., Vandenbogaert, M., Weng, Z., Workman, C., Ye, C., Zhu, Z.: Assessing computational tools for the discovery of transcription factor binding sites. Nat. Biotechnol. 23, 137–144 (2005)
50. Tu, X., Latifi, T., Bougdour, A., Gottesman, S., Groisman, E.A.: The PhoP/PhoQ two-component system stabilizes the alternative sigma factor RpoS in Salmonella enterica. Proc. Natl. Acad. Sci. USA 103, 13503–13508 (2006)
51. Winfield, M.D., Groisman, E.A.: Phenotypic differences between Salmonella and Escherichia coli resulting from the disparate regulation of homologous genes. Proc. Natl. Acad. Sci. USA 101, 17162–17167 (2004)
52. Ye, Z.: Artificial-intelligence approach for biomedical sample characterization using Raman spectroscopy. IEEE Transactions on Automation Science and Engineering 2, 67–73 (2005)
53. Zwir, I., Shin, D., Kato, A., Nishino, K., Latifi, T., Solomon, F., Hare, J.M., Huang, H., Groisman, E.A.: Dissecting the PhoP regulatory network of Escherichia coli and Salmonella enterica. Proc. Natl. Acad. Sci. USA 102, 2862–2867 (2005)
54. Zwir, I., Zaliz, R.R., Ruspini, E.H.: Automated biological sequence description by genetic multiobjective generalized clustering. Ann. N. Y. Acad. Sci. 980, 65–82 (2002)

Appendix

Tables and supplemental figures are available online at
http://gps-tools2.wustl.edu/FSB/Appendix.pdf

4

Fuzzy Vector Filters for cDNA Microarray Image Processing

Rastislav Lukac[1] and Konstantinos N. Plataniotis[2]

[1] Epson Edge, 3771 Victoria Park Avenue, Toronto, Ontario, M1W 3Z5, Canada
lukacr@ieee.org
http://www.colorimageprocessing.com
[2] The Edward S. Rogers Sr. Department of ECE, University of Toronto, 10 King's College Road, Toronto, Ontario, M5S 3G4, Canada

Summary. A data-adaptive fuzzy filtering framework is designed to remove noise in microarray images without the requirement for fuzzy rules and local statistics estimation, or under unrealistic assumptions that the original signal is available. This is achieved by utilizing the inference engine in the form of transformed distance metrics between the samples within the supporting window. The training of the filter coefficients is thus based on local image features. Proposed fuzzy filters can preserve important structural elements and eliminate degradations introduced during microarray image formation.

4.1 Introduction

Recent technological advances have allowed for the combination of various biological, medical and computational approaches and their application to the field of computational biology, genomic engineering and bioinformatics. Complementary Deoxyribonucleic Acid (cDNA) microarray imaging is one of such advanced technologies [2, 5]. It is used in the investigation of toxicological problems and extraction and interpretation of genomic information via cellular response to low dose ionizing radiation. Analysis of cDNA microarray data helps in monitoring the expression levels of thousands of genes simultaneously [9, 27]. By analyzing changes in genome-wide patterns of gene expression in different populations of cells, potentially hazardous substances, such as carcinogens and reproductive toxins, can be identified. Due to the parallel processing feature and effectiveness of their analysis, cDNA microarrays have found applications in gene and drug discovery, toxicological research, and cancer, diabetes and genetic disease diagnosis [14, 42].

Unfortunately, the often result of microarray imaging are images which suffer from significant image background variations, discrete artifacts, and noise floor [1, 20, 26]. Therefore, extensive image processing is usually necessary in order to eliminate errors from propagating further down the processing pipeline to the gene expression analysis tasks. The vast volume of microarray data necessitates the use of automated image processing. Typical image processing operations include filtering and enhancement [1, 20, 39], edge detection [17, 25],

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 67–82.
springerlink.com

data normalization [18, 38], background separation [28], grid adjustment [6], and image segmentation [8, 12, 15, 19, 23]. Among these, noise filtering is defacto a default element of the processing pipeline, as removing noise in microarray images makes them easier to analyze and gene expression measurements obtained in the end of the process are more accurate to interpret [42]. Taking into consideration the noise characteristics in cDNA microarray images, fuzzy logic-based techniques have been proved to be an effective solution to the estimation problem at hand [21].

This chapter focuses on the design of fuzzy logic-based noise removal techniques for cDNA microarray images. Since in microarray imaging the original noise-free signal is not available to the designer, the problem of determining the optimal filtering structure becomes quite challenging. To achieve the desired processing accuracy, the filtering framework presented in this chapter utilizes weighting coefficients which are adaptively determined on the basis of local signal context expressed via aggregated distances between the inputs. This framework integrates well-known concepts from the areas of fuzzy set theory, nonlinear filtering, multidimensional scaling and robust order-statistics.

Section 4.2 discusses the biological background and microarray basics. Section 4.3 describes the cDNA microarray imaging fundamentals, image representation, and noise impairments. Section 4.4 is devoted to the design of a generalized framework for filtering noise in cDNA microarrays using fuzzy logic principles. The chapter concludes with Section 4.5 by summarizing the main fuzzy logic-based cDNA microarray image filtering ideas.

4.2 Microarray Basics

The microarray experiment usually consists of the following four stages: i) probe design and microarray fabrication, ii) sample preparation and target sequence hybridization, iii) hybridization result detection, and iv) hybridization image analysis. As shown in Fig. 4.1, the cDNA microarray imaging procedure requires first to isolate Ribonucleic Acid (RNA) from both control (reference) and experimental (test) sample in order to convert these extracted RNAs into cDNAs by the so-called reverse transcription process [40]. Using a Cy3/Cy5 system, the procedure continues by labelling the cDNAs with fluorescent probes, usually Cy3 for the control and Cy5 for the experimental channel [26], and hybridizing the fluorescent targets to the microarray, which is an arrays of cDNA spots, usually up to 80 000 probes per 2×4 cm^2 area [5]. After heating microarrays at 65°C and washing them for 16 to 24 hours, microarrays are scanned using a sophisticated scanner.

The scanner excites the fluorescent dyes on the hybridized cDNA samples in order to emit fluorescence photons. Common microarray imaging systems use excitation radiation of a narrow-band (i.e., laser) or wide-band (i.e., lamp) spectrum light source to generate excitation photons. The absorption of the excitation light by the fluorescent dyes results in fluorescence, a process which raises an energy level of the fluorescent molecules to an unstable excited state. When

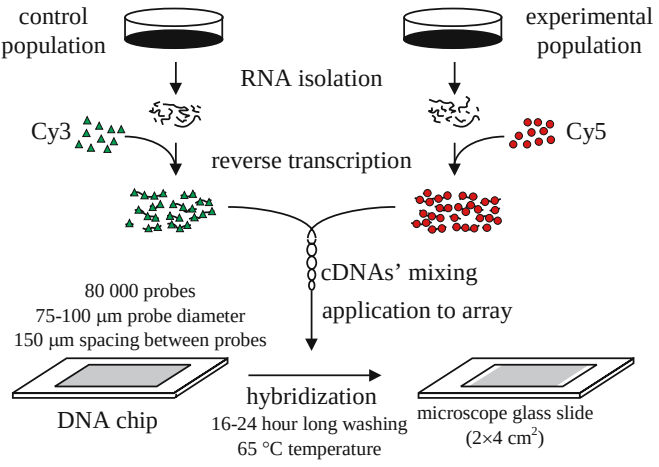


Fig. 4.1. Microarray experiment using Cy3 and Cy5 fluorescent dyes

molecules decay from this state, they emit fluorescent light at a characteristic wavelength (slightly larger than the wavelength of the excitation light [41]), a process which is known as fluorescence emission. Common fluorophores have a very small difference, the so-called Stokes shift [3], between excitation and emission peaks. Namely, as depicted in Fig. 4.2, Cy3 reaches peak absorption at 554 nm and emission at 568 nm, while the corresponding wavelength for Cy5 are 650 nm for peak absorption and 672 nm for emission. To prevent distortion, excitation and emission photons are discriminated by using a dichroic beam-splitter in conjunction with a band-pass optical filter [7, 41]. Since the emitted photons

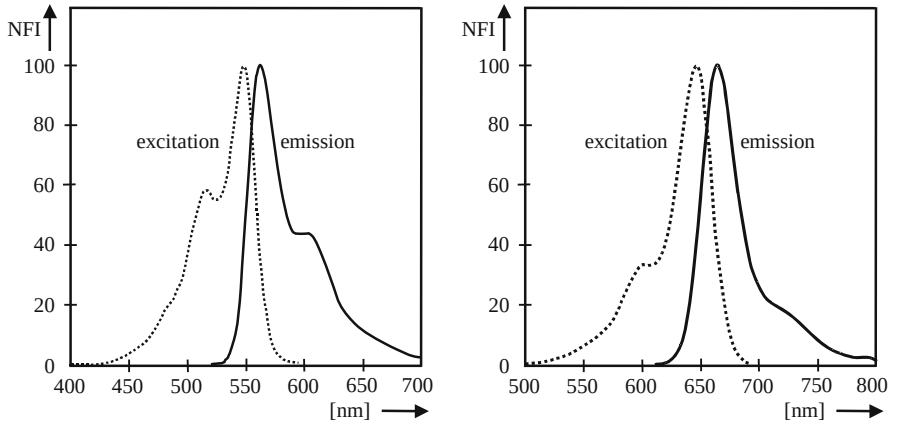


Fig. 4.2. Cyanine dye spectra and the normalized fluorescence intensity (NFI) expressed using the light wavelength: (left) Cy3 and (right) Cy5

vary in their direction, emission optics collect the emitted fluorescence and direct it toward a detector [34], such as a photomultiplier tube (PMT) for laser excitation or a charge-coupled device (CCD) detector for filtered white-light excitation, which detects the emission radiation and converts it into voltage. The scanning process converts the emission photons into electrons. Finally, an analog-to-digital (A/D) converter is used to transform the electric current into a digital signal which is stored as an image formed by equally spaced pixels [29, 41].

4.3 cDNA Microarray Imaging

The scanning procedure produces two 16-bit monochromatic images, one (Fig. 4.3a) for Cy5 and one (Fig. 4.3b) for Cy3. These monochromatic images are registered into a two-channel, red-green image for the purpose of image processing and gene expression analysis. However, according to the trichromatic theory of color vision, an arbitrary color is matched by superimposing appropriate amounts of three primary colors [32]. Therefore, in order to visualize or store cDNA image data in the familiar red-green-blue (RGB) color format (Fig. 4.3c), the introduction of zero B components is needed [20].

The foreground of microarray images is constituted by their spots. The intensities of the pixels within a microarray spot are used to determine a single gene expression and to identify the genes expressed in a particular cell type [10, 42]. A gap between spots or alternatively the presence of cDNA vectors residing outside spots areas constitute the background. By extracting the spots from the microarray image, the background can be viewed as a homogeneous region, while the essential foreground should remain heterogonous as a result of the variable spots' coloration [23].

The image spots' coloration represents the abundance of hybridized RNA in the array [40]. The presence of RNAs from the experimental or control population of cells is determined by the red or green spots, respectively. The occurrence of yellow spots suggests that RNAs from both experimental and control

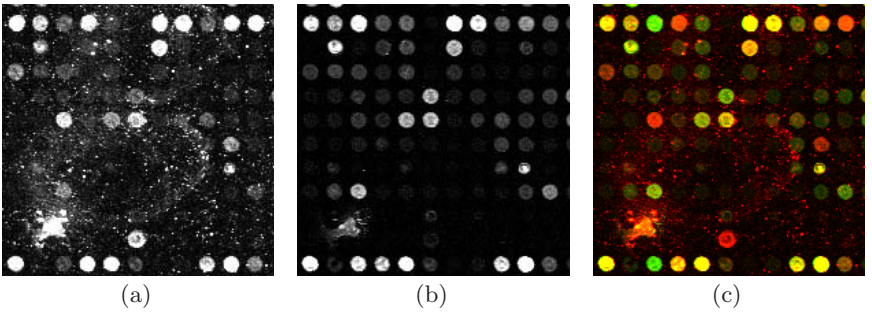


Fig. 4.3. cDNA microarray image: (a) Cy5 channel, (b) Cy3 channel, and (c) cDNA microarray visualized as the RGB image with zero B components

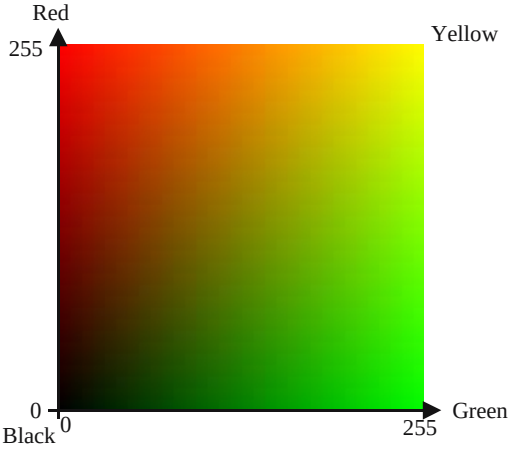


Fig. 4.4. Color mixing in cDNA microarray images

population contribute to the abundance while black spots denote no binding of RNAs. Based on this simple coloration concept (Fig. 4.4), cDNA microarray based gene expression analysis uses the measurement of the hybridized RNA abundance as a measure of gene expression activity [15].

4.3.1 Image Representation

The two-channel cDNA microarray image $\mathbf{x} : Z^2 \rightarrow Z^2$ represents a $K_1 \times K_2$ matrix of two-component samples $\mathbf{x}_{(r,s)} = [x_{(r,s)1}, x_{(r,s)2}]^T$ with $r = 1, 2, ..., K_1$ and $s = 1, 2, ..., K_2$ denoting the image rows and columns, respectively. The component $x_{(r,s)1}$ indicates the R channel while $x_{(r,s)2}$ indicates the G channel, which are combined to form the cDNA vector $\mathbf{x}_{(r,s)}$ uniquely determined by its magnitude $M_{(r,s)} = \left\|\mathbf{x}_{(r,s)}\right\| = \sqrt{(x_{(r,s)1})^2 + (x_{(r,s)2})^2}$ and direction $D_{(r,s)} = \frac{1}{\left\|\mathbf{x}_{(r,s)}\right\|}\mathbf{x}_{(r,s)} = \frac{1}{M_{(r,s)}}\mathbf{x}_{(r,s)}$ in a two-dimensional vector space depicted in Fig. 4.4, [22]. A typical spot has a circular shape and contains approximately 150 to 200 cDNA pixels $\mathbf{x}_{(r,s)}$ [37]. Under the ideal conditions, each spot in the noise-free cDNA image (Fig. 4.5a) has uniform magnitude (Fig. 4.5b) and directional (Fig. 4.5c) characteristics [23].

4.3.2 cDNA Microarray Image Noise

Unfortunately, microarray image formation is a complicated, nonlinear process influenced by many factors resulting in images which usually exhibit significant variations in both their foreground and background due to noise impairments [21, 39]. Main sources of noise in microarray images are artifacts caused by laser light reflection and dust on the glass slide, photon and electronic noise introduced during scanning, and the nature of cDNA microarray technology itself. Because of these noise impairments, spots not only vary in their magnitude

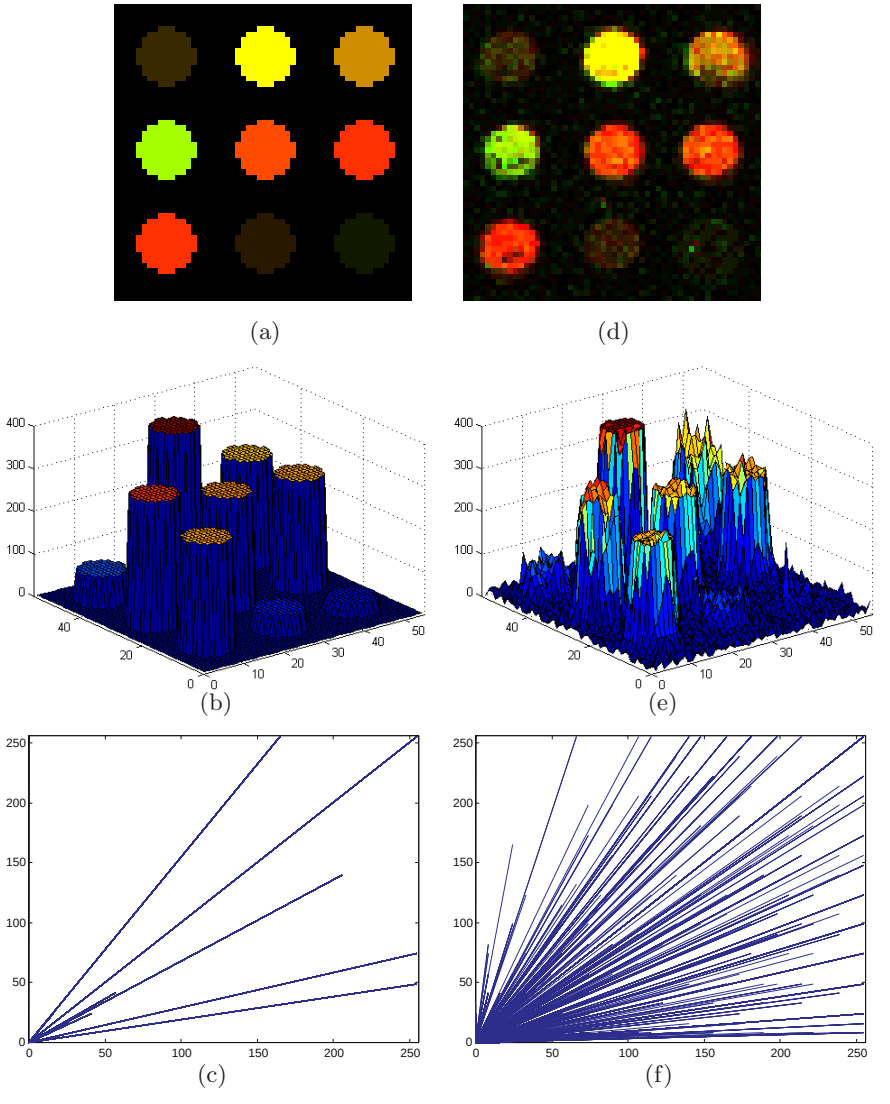


Fig. 4.5. cDNA microarray image characteristics: (a-c) idealized noise-free case, (d-f) real noisy case; (a,d) image area of interest, (b,e) magnitude characteristics, and (c,f) directional characteristics

and directional characteristics (Fig. 4.5d-f), but they often vary in their sizes and positions (Fig. 4.3).

Changes of the pixel intensities from the foreground to the background can be attributed to the Gaussian nature of noise corrupting cDNA chips [26] whereas isolated discrete artifacts and outliers present in the cDNA microarray image can be seen as impairments which are impulsive in nature [20]. In conventional

image processing applications, noise corruption of such nature is most often modelled through a mixture of additive Gaussian noise and impulsive noise [24, 35]. Applying similar modelling concepts to microarray images, the acquired cDNA signal $\mathbf{x}_{(r,s)} = [x_{(r,s)1}, x_{(r,s)2}]^T$ can be expressed as follows:

$$\mathbf{x}_{(r,s)} = \mathbf{o}_{(r,s)} + \mathbf{v}_{(r,s)} \tag{4.1}$$

where $\mathbf{o}_{(r,s)} = [o_{(r,s)1}, o_{(r,s)2}]^T$ represents the original, noise-free cDNA signal and $\mathbf{v}_{(r,s)} = [v_{(r,s)1}, v_{(r,s)2}]^T$ denotes the various image impairments introduced during processing.

4.4 Fuzzy Vector Filtering Framework

The goal of noise filtering in microarray images is to estimate the original image information from noisy data while preserving spot edges and color information as these convey essential information for subsequent analysis. Since cDNA microarray images are nonstationary in their background, suffer from substantial noise floor and impairments, and exhibit significant spot nonuniformities, many filtering schemes operate on the premise that an image can be subdivided into small regions, each of which can be treated as stationary [22]. These small image regions are determined using the supporting window, such as those shown in Fig. 4.6. The window, defined as $\Psi_{(r,s)} = \{\mathbf{x}_{(i,j)}; (i,j) \in \zeta\}$, for $r = 1, 2, ..., K_1$ and $s = 1, 2, ..., K_2$, slides over the entire image $\mathbf{x}$ placing, successively, every pixel at the center of a local neighborhood denoted by ζ. The procedure replaces the cDNA vector $\mathbf{x}_{(r,s)}$ located at the window center (r,s) with the output $\mathbf{y}_{(r,s)} = f(\Psi_{(r,s)})$ of a filter function $f(\cdot)$ operating over the samples listed in $\Psi_{(r,s)}$. As shown in Fig. 4.6, the supporting window may vary in shape. The type of window determines both the area of support and the overall performance of the procedure. Due to its versatility and demonstrated good performance, a 3×3 square-shape window, defined by $\zeta = \{(r+p, s+q); -1 \leq p \leq 1, -1 \leq q \leq 1\}$, is the most commonly used in image processing.

Since it is difficult to distinguish between noise and edge pixels, fuzzy sets — commonly considered as sets with unsharp boundaries — are highly appropriate for image filtering tasks [32]. This is due to the fact that fuzzy sets are better

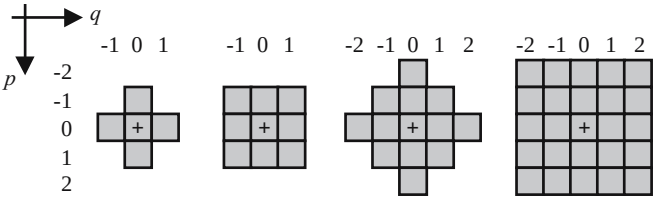


Fig. 4.6. Supporting windows with '+' denoting the center location (r,s). The popular 3×3 window is arranged as the second from the left.

suited to deal with tolerance for some inexactness and imprecision compared to a conventional set theory approach.

4.4.1 Fuzzy Logic Basics

The fuzzy set is defined via the characteristic function $\mu_F : U \rightarrow [0, 1]$, which transforms the elements of U to the fuzzy set with any value, the so-called degree of membership, between 0 and 1. If an element assigns a value close to 1, the degree of membership, or truth value is high. The characteristic function of a fuzzy set is called the membership function and depending on definition it can take a variety of different shapes [44, 45].

A number of fuzzy filters use a window-based, rule-driven approach leading to data-dependent fuzzy solution [4, 33, 36]. As shown in Fig. 4.7, these filters utilize fuzzy logic to convert the linguistic terms into the fuzzy quantities. Namely, the fuzzification procedure transforms the input data into fuzzy values. These are processed in the inference engine using the set of if-then-else fuzzy-rules usually constituted in the if-then format. By applying a bank of fuzzy rules to the fuzzy versions of signal elements which lie within the supporting window, the fuzzy filter yields the filtered output taking into account local characteristics or selected patterns in the neighborhood of the pixel to be processed, thus adapting the filter to local data. The output of the fuzzy filter depends on the fuzzy rule and the defuzzification process which combines the effects of the different rules into an output value which is then converted into the original (crisp) application format. Through the utilization of linguistic terms, a fuzzy rule-based approach to signal processing allows for the incorporation of human knowledge and intuition into the design, which cannot be achieved via traditional mathematical modelling techniques.

Unfortunately, there is no easy way to determine the number and type of fuzzy rules required for the fuzzy image operation. Usually, the designer has to compromise between quality and number of rules used, since even for a moderate supporting window a large number of linguistic rules are required and these rules must be optimally set using the optimization procedure [11, 13] which typically requires the presence of the original signal and the sufficient time for learning. However, as in many other real-life applications, microarray imaging lacks the

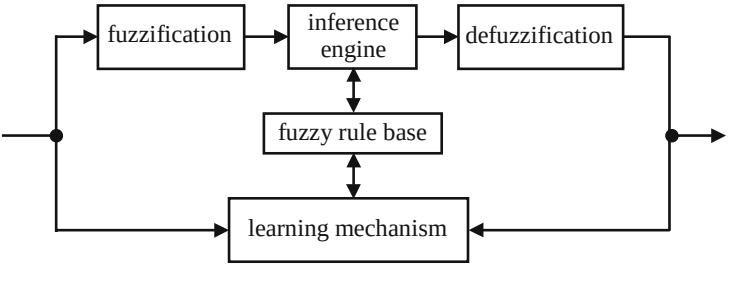


Fig. 4.7. Conventional fuzzy system with the learning mechanism

original signal. To overcome this difficulty, data-dependent filters adopting fuzzy reasoning have been designed for color image processing applications [16, 31, 32]. These designs combine fuzzy concepts, such as membership functions and fuzzy aggregators, with nonlinear estimators. Using similar strategy, fuzzy vector filters have been proposed to remove noise in cDNA microarrays without the requirement for fuzzy rules [21].

4.4.2 Data-Adaptive Fuzzy Filter Design

The most commonly used smoothing method to reduce the level of random noise present in the signal and transitions is averaging. Therefore, the general form of the data-dependent filter is given as a fuzzy weighted average [22, 32] of the cDNA vectors inside the supporting window $\Psi_{(r,s)}$:

$$\mathbf{y}_{(r,s)} = f\left(\sum_{(i,j)\in\zeta} w_{(i,j)}\mathbf{x}_{(i,j)}\right) \tag{4.2}$$

where $f(\cdot)$ is a nonlinear function that operates over the weighted average of the input set and

$$w_{(i,j)} = \mu_{(i,j)} / \sum_{(g,h)\in\zeta} \mu_{(g,h)} \tag{4.3}$$

is the normalized filter weight calculated using the weighting coefficient $\mu_{(i,j)}$ equivalent to the fuzzy membership function associated with the cDNA vector $\mathbf{x}_{(i,j)} \in \Psi_{(r,s)}$. Note that the two constraints $w_{(i,j)} \geq 0$ and $\sum_{(i,j)\in\zeta} w_{(i,j)} = 1$ are necessary to ensure that the filter output is an unbiased estimator and produces the samples within the desired intensity range.

Since noisy samples deviate from other samples in a given data population, outlying cDNA vectors can be determined by evaluating their distance to other

```
Inputs:  K1×K2 input image x
         Supporting window area ζ
         Design parameters α and β
Output:  K1×K2 image y

For r=1 to K1
  For s=1 to K2
    Read the input set Ψ(r,s) = {x(i,j); (i,j) ∈ ς}
    Calculate the aggregated distances D(i,j) using Ψ(r,s)
    Calculate the weights w(i,j) based on D(i,j)
    Determine y(r,s) = f(Ψ(r,s), w, α, β)
  End
End
```

Fig. 4.8. Pseudo-code of data-adaptive fuzzy filtering

vectors in this data population [20]. This rationale can be used to design an effective and computationally efficient filter, as the one shown in Fig. 4.8. The filter weights $\mathbf{w}_{(i,j)}$, for $(i,j) \in \zeta$, can be determined adaptively using functions of a distance criterion between the cDNA vectors included in $\Psi_{(r,s)}$. A membership function value $\mu_{(i,j)}$ can thus be used to quantify the degree of similarity of $\mathbf{x}_{(i,j)}$ to the other cDNA vectors in $\Psi_{(r,s)}$.

Detailed inspection of Fig. 4.5 reveals that the data variations affect both the magnitude and the directionality of the cDNA vectors. Therefore, powerful filtering solutions can be designed by evaluating the vectors' differences in magnitude and/or orientation. For given two cDNA vectors $\mathbf{x}_{(i,j)} = [x_{(i,j)1}, x_{(i,j)2}]^T$ and $\mathbf{x}_{(g,h)} = [x_{(g,h)1}, x_{(g,h)2}]^T$, where $(i,j) \in \zeta$ and $(g,h) \in \zeta$, their magnitude difference can be quantified through the Euclidean metric as follows:

$$d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) = \left(\sum_{k=1}^{2} (x_{(i,j)k} - x_{(g,h)k})^2 \right)^{\frac{1}{2}} \tag{4.4}$$

whereas their orientation difference can be evaluated using the angular measure as follows:

$$d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) = \arccos \left(\frac{\mathbf{x}_{(i,j)} \cdot \mathbf{x}_{(g,h)}}{|\mathbf{x}_{(i,j)}||\mathbf{x}_{(g,h)}|} \right) \tag{4.5}$$

Based on the value of $d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)})$, the corresponding fuzzy membership function is then computable as:

$$\mu(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) = \frac{1}{1 + f(d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}))} \tag{4.6}$$

where $\mu(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) \to 0$ for $d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) \to \infty$ and $\mu(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) = 1$ for $d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) = 0$. Depending on the specific distance measure that is applied to the input data, a different fuzzy membership function can be devised.

4.4.3 Fuzzy Weighting Formulations

As shown in Fig. 4.9, to avoid optimization of fuzzy rules, the data-adaptive fuzzy system can utilize the inference engine in the form of transformed distance metrics between the input cDNA vectors [21]. Since the output of this adaptive fuzzy system depends on local neighborhood ζ, the system is capable of tracking the varying image and noise statistics. The training or learning of the weighting coefficients is only based on local image features without the use of linguistic fuzzy rules or local statistics estimation. By appropriately tuning their membership function the data-adaptive fuzzy filters can be optimized for any noise model [31].

The weight $w_{(i,j)}$, for $(i,j) \in \zeta$, provides the degree to which an input cDNA vector $\mathbf{x}_{(i,j)}$ contributes to the output of the filter. The relationship between the central sample $\mathbf{x}_{(r,s)}$ and its neighbors determined by $\Psi_{(r,s)}$ should be reflected in the decision for the weights of the filter. Since the relationship between distances measured in physical units and perception is generally exponential [31],

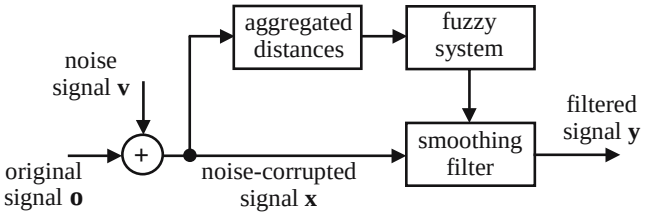


Fig. 4.9. Data-adaptive fuzzy system

an exponential type of function maybe suitable to be used in the weighting formulation:

$$\mu_{(i,j)} = \beta \left(1 + \exp\left\{D_{(i,j)}\right\}\right)^{-\alpha} \tag{4.7}$$

where α is a parameter adjusting the weighting effect of the membership function, β is a normalization constant, and $D_{(i,j)} = \sum_{(g,h)\in\zeta} d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)})$ is the aggregated distance or similarity measure. The data-adaptive filters can be optimized for any noise model by appropriately tuning their membership function in the above equation. These filters can operate either on magnitude or direction of the input cDNA vectors [24]. Depending on the employed distance or similarity function $d(\cdot,\cdot)$, different degrees of membership can be achieved, thus resulting in different filter outputs. This increases the degree of freedom in the filter design.

The defuzzification step is realized via the filtering procedure, which determines the most appropriate vector value to replace the cDNA vector $\mathbf{x}_{(r,s)}$ under consideration and represent a collection of cDNA vectors $\mathbf{x}_{(i,j)}$, for $(i,j) \in \zeta$. A widely used centroid defuzzification approach, the so-called center of gravidity, generates the defuzzified value which is at the center of the values of a fuzzy set [31]. Therefore, the presented data-adaptive filtering scheme based on membership functions defined via the distance concept also satisfies minimization property required in noise removal applications (Fig. 4.10a-c).

The defuzzified vector $\mathbf{y}_{(r,s)}$ obtained through the centroid defuzzification approach does not belong to $\Psi_{(r,s)}$. This suggests that such an unconstrained filter may have reduced detail-preservation ability compared to its constrained version which can operate using the weights defined as follows:

$$w_{(i,j)} = \frac{\mu^{\lambda}_{(i,j)}}{\sum\limits_{(g,h)\in\zeta} \mu^{\lambda}_{(g,h)}} = \frac{\left(\frac{\mu_{(i,j)}}{\mu_{max}}\right)^{\lambda}}{\sum\limits_{(g,h)\in\zeta} \left(\frac{\mu_{(g,h)}}{\mu_{max}}\right)^{\lambda}} \tag{4.8}$$

where $\mu_{max} \in \{\mu_{(i,j)}; (i,j) \in \zeta\}$ is the largest membership value.

Given that $\mu_{(i,j)} < \mu_{max}$ and $\lambda \to \infty$, the weights obtained via the maximum defuzzifier strategy can be redefined as follows [32]:

$$w_{(i,j)} = \begin{cases} 1 & \text{if } \mu_{(i,j)} = \mu_{max} \\ 0 & \text{if } \mu_{(i,j)} \neq \mu_{max} \end{cases} \tag{4.9}$$

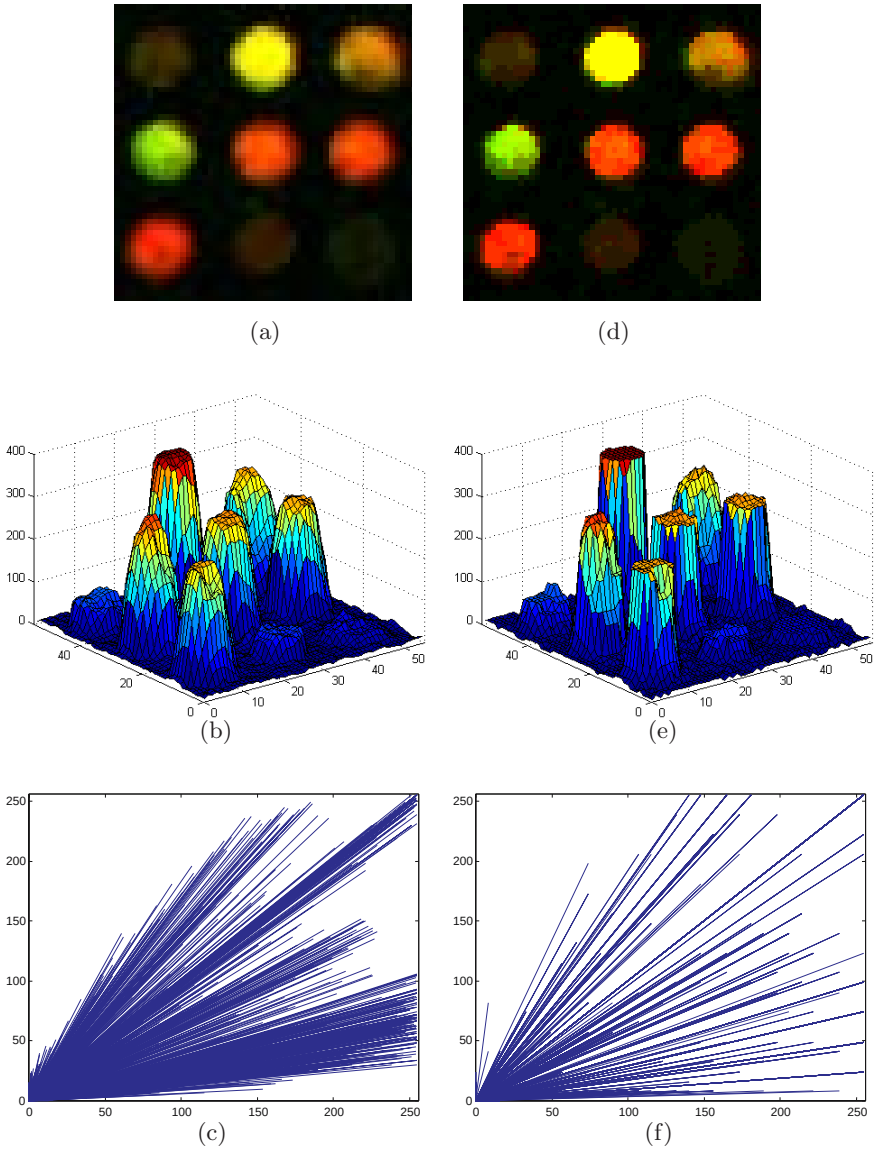


Fig. 4.10. cDNA microarray image characteristics: (a-c) fuzzy weighted averaging vector filter, (d-f) fuzzy selection vector filter; (a,d) image area of interest, (b,e) magnitude characteristics, and (c,f) directional characteristics

If the maximum value occurs at a single point only, the output of an adaptive fuzzy system can be expressed as:

$$\mathbf{y}_{(r,s)} = \mathbf{x}_{(i,j)}, \quad \text{for } \mu_{(i,j)} = \mu_{max} \tag{4.10}$$

which is equivalent to a selection filtering operation which identifies one of the cDNA vectors determined by $\Psi_{(r,s)}$ as the filter output. Such filters are able to preserve both intra- and inter-channel corrections while effectively suppressing noise in image signals (Fig. 4.10d-f).

4.4.4 cDNA Ratio-Based Fuzzy Filter Design

The use of the Euclidean metric or the angular distance in not the only way of evaluating the similarity of two cDNA vectors. Taking advantage of the expected relative uniformity of the local ratios between the components of each cDNA vector, which is definitely the case of noise free-data, aggregated distance $D_{(i,j)}$ can be calculated using $d(\cdot,\cdot)$ expressed as the absolute difference between cDNA ratios [21]:

$$d(\mathbf{x}_{(i,j)}, \mathbf{x}_{(g,h)}) = \left| x_{(i,j)1}/x_{(i,j)2} - x_{(g,h)1}/x_{(g,h)2} \right| \tag{4.11}$$

Operating on the ratios $x_{(i,j)1}/x_{(i,j)2}$, for $(i,j) \in \zeta$, the output cDNA vector $\mathbf{y}_{(r,s)} = [y_{(r,s)1}, y_{(r,s)2}]^T$ is obtained as follows:

$$y_{(r,s)1} = x^*_{(r,s)2} \sum\nolimits_{(i,j)\in\zeta} w_{(i,j)} x_{(i,j)1}/x_{(i,j)2} \tag{4.12}$$

$$y_{(r,s)2} = x^*_{(r,s)1} \sum\nolimits_{(i,j)\in\zeta} w_{(i,j)} (x_{(i,j)1}/x_{(i,j)2})^{-1} \tag{4.13}$$

where $\mathbf{x}^*_{(r,s)} = [x^*_{(r,s)1}, x^*_{(r,s)2}]^T$ is a normalization vector. This vector can be considered as a robust estimate which statistically represents the input set $\Psi_{(r,s)}$. Using robust order-statistic principle [30], $\mathbf{x}^*_{(r,s)}$ is defined here as the componentwise median filter [43]. However, unlike its standard applications where $\mathbf{x}^*_{(r,s)}$ is the output of a filtering procedure, the components of $\mathbf{x}^*_{(r,s)}$ are used here to normalize the output color ratio value to the desired intensity rage in order to recover components $y_{(r,s)1}$ and $y_{(r,s)2}$.

4.5 Conclusion

This chapter presented a data-adaptive fuzzy filtering framework for processing of cDNA microarray images. Unlike conventional fuzzy systems that require complex or time-consuming training procedures and the presence of the original data to optimally set the fuzzy rules in order to achieve the desired performance, the framework described in this chapter is designed to remove noise in cDNA microarray images without the requirement for fuzzy rules or under unrealistic assumptions that the original cDNA signal is available by utilizing the inference engine in the form of transformed distance metrics between the cDNA vector-valued samples within the supporting window. In this way, the training or learning of the weighting coefficients is only based on local image features without the use of linguistic fuzzy rules or local statistics estimation.

Given the vectorial nature of cDNA data and the mixed Gaussian and impulsive nature of noise impairments in microarray images, particular emphasis

was put on powerful vector operators that are proven to be effective and robust for various noise models. By utilizing robust order statistics calculated through a supporting window, fuzzy vector filters can preserve important structural elements, such as spot edges, and eliminate degradations introduced during microarray image formation. The utilization of the spatial, structural and spectral characteristics of the cDNA vector-valued signal is essential in the preservation of both intra- and inter-channel correlations which are of paramount importance for subsequent cDNA microarray analysis.

References

1. Adjeroh, D.A., Zhang, Y., Parthe, R.: On denoising and compression of DNA microarray images. Pattern Recognition 39, 2478–2493 (2006)
2. Ajay, N., Tokuyasu, T., Snijders, A., Segraves, R., Albertson, D., Pinkel, D.: Fully automatic quantification of microarray image data. Genome Research 12, 325–332 (2002)
3. Alfano, R.R., Yang, Y.: Stokes shift emission spectroscopy of human tissue and key biomolecules. IEEE Journal of Selected Topics in Quantum Electronics 9, 48–152 (2003)
4. Arakawa, K.: Median filters based on fuzzy rules and its application to image processing. Fuzzy Sets and Systems 77, 3–13 (1996)
5. Arena, P., Bucolo, M., Fortuna, L., Occhipinty, L.: Celular neural networks for real-time DNA microarray analysis. IEEE Engineering in Medicine and Biology 21, 17–25 (2002)
6. Bajcsy, P.: Gridline: Automatic grid allignment in DNA microarray scans. IEEE Transactions on Image Processing 13, 15–25 (2004)
7. Chen, Y., Dougherty, E., Bittner, M.: Ratio-based decisions and the quantitative analysis cDNA microarray images. Journal of Biomedical Optics 2, 364–374 (1997)
8. Damiance, A.P.G., Zhao, L., Carvalho, A.C.: A dynamic model with adaptive pixel moving for microarray images segmentation. Real-Time Imaging 10, 189–195 (2004)
9. Eisen, M.B., Brown, P.O.: DNA arrays for analysis of gene expression. Methods in Enzymology 303, 179–205 (1999)
10. Filkov, V., Skiena, S., Zhi, J.: Analysis techniques for microarray time-series data. Journal of Computational Biology 9, 317–330 (2002)
11. Goldberg, D.: Genetic algorithms in search, optimisation, and machine learning. Addison-Wesley, Reading (1989)
12. Hirata, R., Barrera, J., Hashimoto, R.F., Dantas, D.O., Esteves, G.H.: Segmentation of microarray images by mathematical morphology. Real-Time Imaging 8, 491–505 (2002)
13. Hou, L.R., Yi, Z.: Fuzzy logic controller based on genetic algorithms. Fuzzy Sets and Systems 83, 1–10 (1996)
14. Istepanian, R.S.H.: Microarray image processing: Current status and future directions. IEEE Transactions on Nanobioscience 2, 173–175 (2003)
15. Katzer, M., Kummert, F., Sagerer, G.: Methods for automatic microarray image segmentation. IEEE Transactions on Nanobioscience 2, 202–213 (2003)
16. Khriji, L., Gabbouj, M.: Adaptive fuzzy order statistics-rational hybrid filters for color image processing. Fuzzy sets and Systems 128, 35–46 (2002)

17. Kim, J.H., Kim, H.Y., Lee, Y.S.: A novel method using edge detection for signal extraction from cDNA microarray image analysis. Experimental and Molecular Medicine 33, 83–88 (2001)
18. Kim, J.H., Shin, D.M., Lee, Y.S.: Effect of local background intensities in the normalization of cDNA microarray data with a skewed expression profiles. Experimental and Molecular Medicine 34, 224–232 (2002)
19. Liew, A.W.C., Yana, H., Yang, M.: Robust adaptive spot segmentation of DNA microarray images. Pattern Recognition 36, 1251–1254 (2003)
20. Lukac, R., Plataniotis, K.N., Smolka, B., Venetsanopoulos, A.N.: A multichannel order-statistic technique for cDNA microarray image processing. IEEE Transactions on Nanobioscience 3, 272–285 (2004)
21. Lukac, R., Plataniotis, K.N., Smolka, B., Venetsanopoulos, A.N.: cDNA microarray image processing using fuzzy vector filtering framework. Journal of Fuzzy Sets and Systems 152, 17–35 (2005)
22. Lukac, R., Smolka, B., Martin, K., Plataniotis, K.N., Venetsanopulos, A.N.: Vector filtering for color imaging. IEEE Signal Processing Magazine 22, 74–86 (2005)
23. Lukac, R., Plataniotis, K.N.: cDNA microarray image segmentation using root signals. International Journal of Imaging Systems and Technology 16, 51–64 (2006)
24. Lukac, R., Plataniotis, K.N.: A taxonomy of color image filtering and enhancement solutions. In: Hawkes, P.W. (ed.) Advances in imaging and electron physics, vol. 140, pp. 187–264. Elsevier / Academic Press, San Diego (2006)
25. Lukac, R., Plataniotis, K.N.: Vector edge operators for cDNA microarray spot localization. Computerized Medical Imaging and Graphics 31, 510–522 (2007)
26. Nagarajan, R.: Intensity-based segmentation of microarrays images. IEEE Transactions on Medical Imaging 22, 882–889 (2003)
27. Nagarajan, R., Upreti, M.: Correlation statistics for cDNA microarray image analysis. IEEE/ACM Transactions in Computational Biology and Bioinformatics 3, 232–238 (2006)
28. O'Neill, P., Magoulas, G.D.: Improved processing of microarray data using image reconstruction techniques. IEEE Transactions on Nanobioscience 2, 176–183 (2003)
29. Pickett, S.C.: Understanding and evaluating fluorescent microarray imaging instruments, IVD Technology (2003)
30. Pitas, I., Venetsanopoulos, A.N.: Order statistics in digital image processing. Proceedings of the IEEE 80, 1892–1919 (1992)
31. Plataniotis, K.N., Androutsos, D., Venetsanopoulos, A.N.: Adaptive fuzzy systems for multichannel signal processing. Proceedings of the IEEE 87, 1601–1622 (1999)
32. Plataniotis, K.N., Venetsanopoulos, A.N.: Color image processing and applications. Springer, Berlin (2000)
33. Russo, F., Ramponi, G.: A fuzzy filter for images corrupted by impulse noise. IEEE Signal Processing Letters 3, 168–170 (1996)
34. Schena, M.: Microarray biochip technology. Eaton Publishing Company / Biotechniques Books (2000)
35. Tang, K., Astola, J., Neuvo, Y.: Nonlinear multivariate image filtering techniques. IEEE Transactions on Image Processing 4, 788–798 (1995)
36. Tsai, H.H., Yu, P.T.: Genetic-based fuzzy hybrid multichannel filters for color image restoration. Fuzzy Sets and Systems 114, 203–224 (2000)
37. Wang, X., Ghosh, S., Guo, S.W.: Quantitative quality control in microarray processing and data acquisition. Nucleic Acids Research 29, 1–8 (2001)
38. Wang, Y., Lu, J., Lee, R., Gu, Z., Clarke, R.: Iterative normalization of CDNA microarray data. IEEE Transactions on Information Technology in Biomedicine 6, 29–37 (2002)

39. Wang, X.H., Istepian, R.S.H., Song, Y.H.: Microarray image enhancement using stationary wavelet transform. IEEE Transactions on Nanobioscience 2, 184–189 (2003)
40. Whitchurch, A.K.: Gene expression microarrays. IEEE Potentials 21, 30–34 (2002)
41. Yang, Y.H., Buckley, M.J., Dudoit, S., Speed, T.P.: Comparison of methods for image analysis on cDNA microarray data. Journal of Computational and Graphical Statistics 11, 108–136 (2002)
42. Zhang, X.Y., Chen, F., Zhang, Y.T., Agner, S.G., Akay, M., Lu, Z.H., Waye, M.M.Y., Tsui, S.K.W.: Signal processing techniques in genomic engineering. Proceedings of the IEEE 90, 1822–1833 (2002)
43. Zheng, J., Valavanis, K.P., Gauch, J.M.: Noise removal from color images. Journal of Intelligent and Robotic Systems 7, 257–285 (1993)
44. Zilouchian, A., Jamshidi, M.: Intelligent control systems using soft computing methodology. CRC Press, Boca Raton (2001)
45. Zimmermann, H.: Fuzzy set theory and its applications, 2nd edn. Kluwer Academic Publishers, Dordrecht (1991)

5

Microarray Data Analysis Using Fuzzy Clustering Algorithms

Doulaye Dembélé

IGBMC, CNRS-INSERM-ULP
1 rue Laurent Fries, BP 10142
67404 Illkirch Cedex, France
doulaye@titus.u-strasbg.fr

Summary. Microarray technology is used for studying gene regulation at the genome and transcriptome level. In the most common application, the expression level of thousands of genes is monitored simultaneously leading to a huge dataset having high dimensionality. It is assumed that genes with similar function or regulatory elements will display a common expression profile over a variety of biological conditions. For some cases, it may be desirable to study simultaneously many drugs in different experimental conditions (e.g. concentration or time point) on biological models, leading to the generation of 3-way data. Cluster analysis is used for identifying biologically relevant groups of genes. In this chapter, fuzzy cluster analysis is used for this purpose. After a brief formulation of the problem, we outline motivations for our choice of the clustering algorithm. Then, the fuzzy clustering algorithms are presented and the main tuning parameters are discussed in the context of 2-way and 3-way microarray data. We propose a transformation allowing more contrast in distances between all pairs of samples in a dataset. This increases the likelihood of detection of a group structure, if any, in a high dimensional dataset. Results showing the performance of the fuzzy C-Means algorithm are carried out using real datasets. These results are finally validated through functional enrichment of genes.

5.1 Introduction

Microarray technology allows simultaneous monitoring of the expression level of thousands of genes. This technology is actually used routinely in biomedical research to compare gene expression levels at different developmental stages [21], in different tissues [2, 15] or different clinical conditions [10]. Microarrays are solid supports (glass slides) on which thousands of DNA sequences representing genes are spotted (or synthesized in situ) at known addresses (spots). Messenger RNA from biological samples are labelled with a fluorescent dye and hybridized for pairing with complementary DNA sequences on the microarray. The microarray is then washed and scanned to quantify fluorescent molecules hybridized to each DNA sequence. Data from scanned images are first normalized [32, 8] to eliminate variations which are independent of the biological phenomenon studied. After data normalization and/or summarization, genes that best fit the biological model are identified using heuristic or statistical tests [33, 9]. These genes

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 83–102.
springerlink.com

are finally used in classification/clustering algorithms and in data interpretation tools [31, 27].

A typical microarray experiment will for example compare gene expression profiles between multiple biological samples such as tumour biopsies, or a single sample in response to a treatment over time. The dataset produced by such experiments is a table containing thousands of genes (equal to the number of rows), and dozens of samples or arrays (equal to the number of columns). It is generally assumed that genes with similar function or sharing regulatory elements will display a common expression profile over a variety of biological conditions. Classification methods are then used to group genes according to their expression profile in a defined set of samples and/or to group samples based on the set of genes they express [2, 15, 3]. Unsupervised classification methods or clustering are typically used when no *a priori* information is available on samples or genes [13, 4, 28, 29]. There are roughly two kinds of clustering methods [30] : hierarchical and partitional methods. Hierarchical clustering methods produce results represented by dendrograms, like trees where each branch is a group of genes having similar profile. In the agglomerative hierarchical algorithm, each gene is initially put in its own group and an aggregation measure is chosen. Then, the two closest groups are put together into one group. This process is repeated until all genes are in a single group. The divisive hierarchical method operates in the reverse manner, starting with all genes in the same group, it ends with each gene in its own group. The use of the hierarchical method for microarray data clustering has been popularized by Eisen et al [13] who also provided free software for visualizing the results. With a hierarchical method the user does not have to fix the number of clusters *a priori*. However, this method suffers from the non-uniqueness of the dendrogram produced [24].

Partitional clustering methods consist of finding the best partition of genes into K clusters in such a way that one criterion (e.g. total inertia of clusters) is optimized. An exhaustive solution will consist in testing all combinations corresponding to the distribution of genes in the dataset into K clusters and then keeping the combination which minimizes the chosen criterion. In practice, an exhaustive search is rarely performed because of the heavy computation it requires. Instead, heuristic iterative procedures are used. Typically, data are initially (randomly) divided into K clusters. Then iteratively, the best local combination of genes into K clusters is searched by finding in the neighbourhood of each gene, the cluster for which the criterion is optimized. The K-Means algorithm is the most common partitional clustering method. It allows the placement of each gene into one group containing genes of similar profile. Genes in the same cluster are then expected to have similar biological function. With the K-Means clustering algorithm, every gene in the dataset will be assigned to a cluster even if it has a profile different from those of the other genes. This hard assignation of genes to groups was used and gave acceptable results in many cases [29]. However the biological activities of a gene are more complex. It is indeed known that given genes are subject to regulation by several molecular pathways. The overall expression pattern for a given gene may therefore correspond to the superimposition of distinct patterns, each

corresponding to a given mode of regulation. To try to capture this complexity and to be able to form tightly related groups of genes, we used Fuzzy C-Means (FCM) clustering algorithm. In contrast to the K-Means clustering algorithm, the FCM clustering algorithm links each gene to all clusters via a real-valued vector u_{ik} of indexes. The value of components of this vector of indexes are between 0 and 1. The closer an index is to 1 for a given cluster, the closer the corresponding gene is to that cluster. The real-valued vector of indexes defines the membership of a gene to a cluster. Exploitation of membership values will hence identify genes that are tightly related, or genes which are linked to more than one cluster, thus providing the opportunity to explore the biological complexity of a gene.

5.2 Fuzzy C-Means Clustering Algorithm

Let us consider a microarray experiment which consists in studying the expresion profiles of genes for the mouse genome in many biological conditions. Let us note by I the total number of genes in the mouse genome and J the total number of biological conditions of interest. We hence have a *2-way* dataset noted $X = \{\mathbf{x}_1\ \mathbf{x}_2\ \dots\ \mathbf{x}_I\}$ where $\mathbf{x}_i\ (i = 1, \dots, I)$ denotes a vector of size J representing gene i with the measurements for the J biological conditions used.

5.2.1 FCM for 2-Way Microarray Data

Grouping genes in dataset X into K clusters using the FCM algorithm consists in finding cluster membership values u_{ik} and centers $\mathbf{c}_k$ which minimize the criterion proposed in [7, page 65]. This criterion is modified as follows for introducing a transformation of distances used:

$$\mathcal{J}(u_{ik}, \mathbf{c}_k) = \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik}^m d^\alpha(\mathbf{x}_i, \mathbf{c}_k) \tag{5.1}$$

$$subject\ to \quad \sum_{k=1}^{K} u_{ik} = 1 \quad ; \quad 1 \leq i \leq I \tag{5.2}$$

$$with \quad d^\alpha(\mathbf{x}_i, \mathbf{c}_k) = \left(\sum_{j=1}^{J} (x_{ij} - c_{kj})^2 \right)^{\frac{\alpha}{2}} \tag{5.3}$$

where m is the fuzziness parameter ($m > 1$) and $d^\alpha(\mathbf{x}_i, \mathbf{c}_k)$ is $\alpha-$power of the Euclidean distance between gene $\mathbf{x}_i$ and centroid $\mathbf{c}_k$. Usually the Euclidean distance is used thanks to its nice geometrical properties. For other choices of distance, see [17, 14, 22]. The choice of the fuzzy parameter m is discussed in [11]. The use of parameter α in Equation (5.1) is new and it allows more freedom in the choice of parameter m, see Section 5.2.3. We assumed that the number K of clusters in the dataset is known. The value of K can be estimated using ad hoc methods or clustering validation methods [23, 26, 34].

From Equation (5.1), when the number K of clusters, the fuzziness parameter m and the distance power α are fixed the parameters of interest are the cluster membership values u_{ik} and the centers $\mathbf{c}_k$. The total number of parameters to obtain in minimization of criterion (5.1) is $(I + J)K$. This number is high and a direct optimization method is rarely used. Instead, alternating optimization over membership values and centers is used. This algorithm is summarized as followed:

1. Initialization: set values for m, K, α, $\mathbf{c}_k$ and choose a distance metric $d(,)$
2. compute membership values u_{ik}
3. compute cluster centers $\mathbf{c}_k$
4. if convergence stop else goto step 2

Expressions for membership values and cluster centers are obtained as follows:

Expression for the membership values

By using the Lagrange multipliers to take the constraint (5.2) into account in Equation (5.1), we get

$$\mathcal{L}(u_{ik}, \lambda_i) = \sum_{i=1}^{I}\sum_{k=1}^{K} u_{ik}^m d^\alpha(\mathbf{x}_i, \mathbf{c}_k) - \sum_{i=1}^{I} \lambda_i \left(\sum_{k=1}^{K} u_{ik} - 1 \right) \tag{5.4}$$

Performing partial differentiation of Equation (5.4) with respect to u_{ik} we get

$$\frac{\partial \mathcal{L}(u_{ik}, \lambda_i)}{\partial u_{rs}} = m u_{rs}^{(m-1)} d^\alpha(\mathbf{x}_r^t, \mathbf{c}_s^t) - \lambda_r \tag{5.5}$$

setting this derivative to zero, saddle point condition, we get an expression for u_{ik}

$$\frac{\partial \mathcal{L}(u_{ik}, \lambda_i)}{\partial u_{rs}} = 0 \Rightarrow u_{rs} = \frac{\lambda_r^{\frac{1}{m-1}}}{[m d^\alpha(\mathbf{x}_r, \mathbf{c}_s)]^{\frac{1}{m-1}}} \tag{5.6}$$

Performing the partial differentiation of Equation (5.4) with respect to λ_i and setting this derivative to zero, we get

$$\frac{\partial \mathcal{L}(u_{ik}, \lambda_i)}{\partial \lambda_r} = 0 \Rightarrow \sum_{s=1}^{K} u_{rs} = 1 \tag{5.7}$$

Using Equation (5.6) in Equation (5.7) we get:

$$\sum_{k=1}^{K} \frac{\lambda_r^{\frac{1}{(m-1)}}}{[m d^\alpha(\mathbf{x}_r, \mathbf{c}_k)]^{\frac{1}{(m-1)}}} = 1 \tag{5.8}$$

From this equation we have the following expression for λ_i:

$$\lambda_r^{\frac{1}{(m-1)}} = \frac{1}{\sum_{k=1}^{K} \left[\frac{1}{m d^\alpha(\mathbf{x}_r, \mathbf{c}_k)} \right]^{\frac{1}{m-1}}} \tag{5.9}$$

By returning the above expression of λ_i in Equation (5.6) we get

$$u_{rs} = \frac{1}{\sum_{k=1}^{K} \frac{[md^{\alpha}(\mathbf{x}_r, \mathbf{c}_s)]^{\frac{1}{m-1}}}{[md^{\alpha}(\mathbf{x}_r, \mathbf{c}_k)]^{\frac{1}{m-1}}}} \tag{5.10}$$

or

$$u_{rs} = \frac{[d^{\alpha}(\mathbf{x}_r, \mathbf{c}_s)]^{\frac{1}{1-m}}}{\sum_{k=1}^{K} [d^{\alpha}(\mathbf{x}_r, \mathbf{c}_k)]^{\frac{1}{1-m}}} \tag{5.11}$$

Expression for cluster centers

For the cluster centers let us perfom partial differentiation of Equation (5.1) and set it to zero. We get:

$$\frac{\partial \mathcal{J}(u_{ik}, \mathbf{c}_k)}{\partial \mathbf{c}_k} = 0 \Rightarrow \sum_{i=1}^{I} u_{ik}^{m} \frac{\partial d^{\alpha}(\mathbf{x}_i, \mathbf{c}_k)}{\partial \mathbf{c}_k} = 0 \tag{5.12}$$

Using Equation (5.3), we have:

$$\alpha \sum_{i=1}^{I} u_{ik}^{m} \left(\sum_{j=1}^{J} (x_{ij} - c_{kj})^2 \right)^{\frac{\alpha}{2}-1} (x_{is} - c_{ks}) = 0 \tag{5.13}$$

When $\alpha = 2$, i.e. we used the square of Euclidean distance, it follows immediately that the cluster centers are given by

$$c_{ks} = \frac{\sum_{i=1}^{I} u_{ik}^{m} x_{is}}{\sum_{i=1}^{I} u_{ik}^{m}} \quad s = 1, \ldots, J \tag{5.14}$$

In fact, Equation (5.14) is also the solution for all α since all components in the sum $\sum_{j=1}^{J} (x_{ij} - c_{kj})^2$ are greater than or equal to zero with strict equality when $x_{ij} = c_{kj}$ for $j = 1 \ldots, J$.

The alternating optimization over membership values and cluster centers is then based on Equations (5.11) and (5.14). For computing the membership values using Equation (5.11) we need a precaution to deal with the case where cluster centers match some dataset samples ($\mathbf{x}_i = \mathbf{c}_k$). In this case, let us note by $\mathcal{I}_i = \{k / 1 \leq k \leq K \ ; \ d(\mathbf{x}_i, \mathbf{c}_k) = 0\}$ and $\tilde{\mathcal{I}}_i = \{1, 2, \ldots, K\} - \mathcal{I}_i$. Hence, if the set $\mathcal{I}_i$ is empty, we used Equation (5.11), else membership values are set to zero if $k \in \tilde{\mathcal{I}}_i$ and the other values are chosen in such a way that $\sum_{k \in \mathcal{I}_i}^{K} u_{ik} = 1$.

For the convergence, a difference matrix is computed using the matrix associated to membership values ($U = [u_{ik}], \ i = 1, \ldots, I; \ k = 1, \ldots, K$) from the current and previous iteration. The algorithm converges when the Frobenius norm of the difference matrix is lower than a threshold fixed *a priori*. Global convergence results of the FCM algorithm are given in [18, 16].

5.2.2 FCM for 3-Way Microarray Data

When the microarray experiment described in the beginning of this section is performed at T time points, we obtain a *3-way* dataset which can be noted $X_t = \{\mathbf{x}_{1t}, \mathbf{x}_{2t}, \ldots, \mathbf{x}_{It}\}$. We can consider data at each time point as a *2-way* dataset and hence use the objective in Equation (5.1) to get membership values and centers for clusters. In this way, unrelated membership values and centers are produced. When we want to have the same membership values for all I genes independently of the time point, we have to perform simultaneous minimization of T criteria:

$$\min\left(\mathcal{J}_1(u_{ik}, \mathbf{c}_{k1}),\ \mathcal{J}_2(u_{ik}, \mathbf{c}_{k2}),\ \ldots,\ \mathcal{J}_T(u_{ik}, \mathbf{c}_{kT})\right) \tag{5.15}$$

Cluster centers, $\mathbf{c}_{kt}$, $t = 1, \ldots, T$, obtained from Equation (5.15) are time point t dependent. This is not the case for the membership values u_{ik} which are mutually related for all data time points.

The minimization of Equation (5.15) is referred to as a multi-objective optimization problem where managing conflicting and incommensurate objectives can occur. We need here, the membership values u_{ik} and cluster centers $\mathbf{c}_{kt}$ leading simultaneously to the minimum value for each of the T objectives $\mathcal{J}_t(u_{ik}, \mathbf{c}_{kt})$, $t = 1, \ldots, T$. In general, such a solution does not exist and we are interested to all solutions such that an improvement in minimization of one criterion leads to a deterioriation in performances for at least another criterion. These solutions are known as Pareto optimal front in multi-objective optimization. To solve the minimization problem in Equation (5.15), we combined the T objectives into one scalar objective through weights: w_t, $t = 1, \ldots, T$. The total criterion to minimize is then given by the following equation which is an extension of criterion in Equation (5.1), see also [25]:

$$\mathcal{J}(u_{ik}, \mathbf{C}_k) = \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{k=1}^{K} w_t u_{ik}^m d^\alpha(\mathbf{x}_{it}, \mathbf{c}_{kt}) \tag{5.16}$$

subject to the constraint given in Equation (5.2). We assumed that there is no empty cluster. In Equation (5.16), we note $\mathbf{C}_k = [\mathbf{c}_{k1}\ \mathbf{c}_{k2}\ \ldots\ \mathbf{c}_{kT}]$.

The method of combining the T objectives necessitates the choice of T weights. A combination of weights will allow one solution for Equation (5.16), i.e. one point of the Pareto optimal front. The choice for the weights is not trivial. However, using a constraint, $\sum_{t=1}^{T} w_t = 1$ allows the use of many combinations for getting the entire Pareto optimal front. But only a convex curve will be obtained with this approach.

As for a *2-way* dataset case, an alternating optimization algorithm can be used to obtain the cluster membership values and centroids. By mimicking the results in Section 5.2.1, we get the following algorithm, see also [25] where the same equations are obtained using a different method:

Algorithm

Computation of u_{ik} when cluster centers $\mathbf{c}_{kt}$ are given

$$\mathcal{I}_i = \{k/1 \leq k \leq K;\ d^{\alpha}(\mathbf{x}_{it}, \mathbf{c}_{kt}) = 0\} \tag{5.17}$$

$$\tilde{\mathcal{I}}_i = \{1,\ 2,\ \ldots,\ K\} - \mathcal{I}_i \tag{5.18}$$

If $\mathcal{I}_i = \phi$ (an empty set) then

$$u_{rs} = \frac{\left[\sum_{t=1}^{T} w_t d^{\alpha}(\mathbf{x}_{rt}, \mathbf{c}_{st})\right]^{\frac{1}{1-m}}}{\sum_{k=1}^{K}\left[\sum_{t=1}^{T} w_t d^{\alpha}(\mathbf{x}_{rt}, \mathbf{c}_{kt})\right]^{\frac{1}{1-m}}} \tag{5.19}$$

If $\mathcal{I}_i \neq \phi$ (not an empty set) then

$$u_{ik} = 0 \ \ \forall k \in \tilde{\mathcal{I}}_i, \ \ \sum_{k \in \mathcal{I}_i}^{K} u_{ik} = 1 \tag{5.20}$$

Computation of $\mathbf{c}_{kt}$ *when membership values* u_{ik} *are given*

$$c_{kjt} = \frac{\sum_{i=1}^{I} u_{ik}^m x_{ijt}}{\sum_{i=1}^{I} u_{ik}^m} \quad (k = 1, \ldots, K;\ j = 1, \ldots, J_t;\ t = 1, \ldots, T) \tag{5.21}$$

5.2.3 Choice of the Distance Power and the Fuzziness Parameter

In the classical FCM algorithm, the square of the Euclidean metric is used ($\alpha = 2$). The choice of the fuzziness parameter m has been discussed in [11] for the high dimensional microarray dataset, and a heuristic method was proposed for computing an upper bound value for the fuzziness parameter m. Here, we give theoretical results supporting our previous choice. Our theoretical results are based on the following assumptions, notations and definitions. We consider the case of a *2-way* dataset and use the square Euclidean metric. However, the result in corollary 1 can be extended to a *3-way* dataset.

- *A1 :* Components of the vector $\mathbf{x}_i$ are independent and identically distributed (i.i.d.) with all absolute moments up to order 4 finite.
- *D1 :* $P[e]$ denotes the probability of event e,
- *D2 :* $E[Y]$ and $var[Y]$ are respectively, the expectation and the variance of the random variable Y,
- *D3 :* $E[x_{ik}^r] = \mu_r(\mathbf{x}_i)$ is the $r-$order moment of component x_{ik}. From assumption *A1* above, this moment is independent of the index k,
- *D4 :* $d(\mathbf{x}_i, \mathbf{x}_l)$ is a non negative real-valued number representing the distance between $\mathbf{x}_i$ and $\mathbf{x}_l$.
- *D5 :* $d_{min} = \min[d^s(\mathbf{x}_i, \mathbf{x}_l)\ l \neq i = 1, \ldots I]$ and $d_{max} = \max[d^s(\mathbf{x}_i, \mathbf{x}_l)\ l \neq i = 1, \ldots I]$ are respectively the minimum and the maximum of the $s-$power distances between all pairs of the I genes in the dataset.

Theorem 1. *[6] Under the conditions of assumption and definitions given above, if*

$$\lim_{J\to\infty} var\left[\frac{d^s(\mathbf{x}_i,\mathbf{x}_l)}{E[d^s(\mathbf{x}_i,\mathbf{x}_l)]}\right]=0 \tag{5.22}$$

then for every $\epsilon>0$

$$\lim_{J\to\infty} P[d_{max}\leq(1+\epsilon)d_{min})]=1 \tag{5.23}$$

Proof. See [6] for details. ⋄

In short, theorem 1 says that if the distribution of distances fulfills the conditions of relation (5.22) as data dimension J increases, distances between all the pairs of genes goes to nearly the same value. As a consequence, membership values for all genes will go to the same value. In fact, let us note by $\mathcal{D}$ the common value of distances when the condition in relation (5.22) is fulfilled. From Equation (5.11) and assuming that the cluster centroids match some genes, we will have $u_{rs}=(\mathcal{D})^{\frac{1}{1-m}}/\sum_{k=1}^{K}(\mathcal{D})^{\frac{1}{1-m}}=\frac{1}{K}$. We also get the same value for the membership values when the fuzziness parameter m goes to infinity even if more contrast is present in the dataset distances. The convergence of membership values to the same value is observed for high but finite dimensional dataset. The following corollary allows a partial explanation to this situation.

Corollary 1. *Let us assume that the cluster centers match some genes in the dataset. Under the conditions of assumptions and definitions given above, the FCM algorithm based on the square of Euclidean metric cannot recover a group structure from the dataset, if any, when their dimension* J *goes to infinity.*

Proof. Using expression (5.3) of distance with $\alpha=2$, let us show that condition in relation (5.22) is fulfilled when J goes to infinity.

We have[1]:

$$d^2(\mathbf{x}_i,\mathbf{x}_l)=\sum_{j=1}^{J}(x_{ij}-x_{lj})^2=\sum_{j=1}^{J}\sum_{r=0}^{2}\binom{2}{r}(-1)^r x_{ij}^{(2-r)}x_{lj}^r \tag{5.24}$$

$$\begin{aligned} d^4(\mathbf{x}_i,\mathbf{x}_l)&=\left[\sum_{j=1}^{J}(x_{ij}-x_{lj})^2\right]\left[\sum_{j=1}^{J}(x_{ij}-x_{lj})^2\right]\\ &=\sum_{j=1}^{J}\sum_{r=0}^{4}\binom{4}{r}(-1)^r x_{ij}^{4-r}x_{lj}^r\\ &\quad+\sum_{j=1}^{J}\sum_{\substack{k=1\\k\neq j}}^{J}\sum_{r=0}^{2}\sum_{s=0}^{2}\binom{2}{r}\binom{2}{s}(-1)^{r+s}x_{ij}^{2-r}x_{lj}^r x_{ik}^{2-s}x_{lk}^s \end{aligned} \tag{5.25}$$

[1] We used : $\binom{n}{p}=\frac{n!}{p!(n-p)!}$ and $(a-b)^q=\sum_{r=0}^{q}\binom{q}{r}(-1)^r a^{q-r}b^r$.

From Equation (5.24) we have[2]:

$$E[d^2(\mathbf{x}_i, \mathbf{x}_l)] = J\left(\sum_{r=0}^{2} \binom{2}{r} (-1)^r \mu_{(2-r)}(\mathbf{x}_i)\mu_r(\mathbf{x}_l)\right) \tag{5.26}$$

From Equation (5.25) and using $var[Y] = E[Y^2] - E^2[Y]$, we have:

$$var[d^2(\mathbf{x}_i, \mathbf{x}_l)] = J\left(\sum_{r=0}^{4} \binom{4}{r} (-1)^r \mu_{(4-r)}(\mathbf{x}_i)\mu_r(\mathbf{x}_l) - \left[\sum_{r=0}^{2} \binom{2}{r} (-1)^r \mu_{(2-r)}(\mathbf{x}_i)\mu_r(\mathbf{x}_l)\right]^2\right) \tag{5.27}$$

Equation (5.22) can be written as follows:

$$\lim_{J\to\infty} var\left[\frac{d^2(\mathbf{x}_i, \mathbf{x}_l)}{E[d^2(\mathbf{x}_i, \mathbf{x}_l)]}\right] = \lim_{J\to\infty} \frac{var[d^2(\mathbf{x}_i, \mathbf{x}_l)]}{E^2[d^2(\mathbf{x}_i, \mathbf{x}_l)]} \tag{5.28}$$

Using Equations (5.26) and (5.27) in Equation (5.28) we get the following:

$$\lim_{J\to\infty} var\left[\frac{d^2(\mathbf{x}_i, \mathbf{x}_l)}{E[d^2(\mathbf{x}_i, \mathbf{x}_l)]}\right] = \frac{1}{J}\left(\frac{\sum_{r=0}^{4} \binom{4}{r} (-1)^r \mu_{(4-r)}(\mathbf{x}_i)\mu_r(\mathbf{x}_l)}{\left[\sum_{r=0}^{2} \binom{2}{r} (-1)^r \mu_{(2-r)}(\mathbf{x}_i)\mu_r(\mathbf{x}_l)\right]^2} - 1\right) \tag{5.29}$$

This function depends on the first fourth-order moments of the dataset X. From the assumptions on the moments, summations in the numerator and the denominator of Equation (5.29) are finite, we then conclude that Equation (5.29) goes to zero when $J \to \infty$, hence the corollary 1. ◇

Corollary 1 allows an explanation, in part, for the worst behavior of classification algorithms based on Euclidean distance, when the data dimension is higher. The convergence to zero of the expression value of Equation (5.29) assumes that the data dimension J goes to infinity, which is not the case for practical situations. The threshold value of J which can be considered as higher is theoretically difficult to estimate. Let us note that the numerator in the right hand side of Equation (5.29) used moments up to order 4 while only up to 2 order moments are used in the denominator. We used, in the sequel of this subsection, a power of distance through parameter α to increase value of the numerator more rapidly than the denominator, and hence to slow down the convergence to zero of Equation (5.29) for given dataset dimension J.

[2] We use the following property : if $f(Y)$ and $g(Z)$ are two measurable functions were Y and Z are independent random variables, then $E[f(Y)g(Z)] = E[f(Y)]E[g(Z)]$.

Choice of the fuzziness parameter m

The fuzziness parameter m is a real-valued number to be chosen greater than 1. More often it is set to two, which leads to simplification in the expression (5.11) used to compute membership values. However, the previous results show that this choice for m, combined with the square of Euclidean metric can fail to allow the recovery of group structures, if any, in a high dimensional dataset. From Equation (5.11), the cluster membership values depends on the $\frac{1}{m-1}$ power of distances between all pairs of genes in the dataset and the cluster centers. We can assume without loss of generality that cluster centers match some genes. Hence, using this observation and the distribution of distances for a given dataset, we were able to propose a method for estimating an upper bound value m_{ub} for the fuzziness parameter [11]. We heuristically searched for m_{ub} such that the coefficient of variation of the distances between all pairs of genes in the dataset are close to $0.03J$. Equation (5.29) shows that the square of the coefficient of variation of distances between all pairs of dataset genes is inversely proportional to the dataset dimension J.

The method proposed in [11] can lead to a fuzzy parameter m close to 1. When the fuzziness parameter goes to 1 the FCM results are close to those obtained using K-Means clustering algorithm, meaning that one membership value of a given gene is close to 1 while others go to zero. To handle this situation, we introduce the parameter α as a power factor for distances.

Choice of the distance power parameter α

Using the algorithm proposed in [11] for estimating an upper bound value for the fuzziness parameter, we can obtain a value close to 1 for square Euclidean distances. Using parameter α introduces more flexibility in the choice of the fuzziness parameter m. Hence, the computation of an upper bound value can be based on the distances defined by $\{[d^{\alpha}(\mathbf{x}_i, \mathbf{x}_l)]^{\frac{1}{m-1}};\ l \neq i = 1, \ldots, I\}$. Note that the transformation of distances through the parameter α preserves the main properties for the measure of distance (non negativity, identity, symmetry and triangle inequality property). α is a real-valued number greater than zero and is a magnification factor. It accentuates or attenuates differences in distances computed from a dataset. In fact, the choice of α will depend on the objective : increase or decrease of the contrast in distances. The parameter α should be chosen greater or lesser than one depending on the maximum value of data distances and on the modification of distances wide (maximum distance - minimim distance) needed. Hence, if distances between all pairs of genes in the dataset are less than one, the choice $\alpha < 1$ leads to transformed distance values to go toward one. The choice of $\alpha > 1$ will lead to transformed distance values to go toward zero. On the other hand, if at least one distance from those between all pairs of genes in the dataset is greater than one : the choice of $\alpha < 1$ leads to transformed distance values to decrease toward zero while the choice of $\alpha > 1$ leads to transformed distance values to increase toward infinity.

5.3 Results

Microarray data used in this paper has been recently published in [1] where a detailed description of the biological aspects are given. These data are available from the Gene Expression Omnibus (GEO) web site (*http://www.ncbi.nlm.nih.gov/geo/*) under the accession number GSE3634. Affymetrix GeneChip *430A 2.0* mouse arrays were used for generating the data which consisted in studying gene expression profiles in mouse models for Huntington's disease and Spinocerebellar ataxia type 7, respectively, sharing a common retinal phenotype. For the Spinocerebellar ataxia type 7, two genotypes R7E Knock-Out (KO) and Wild-Type (WT) mice were studied at two time points, 3 weeks and 9 weeks of age. For the Huntington's disease model, two genotypes R6/2 KO and WT mice were studied at 9 weeks of age. In each biological condition, 4 to 6 replicates were used. Table 5.1 summarizes the experimental design and the repartition of array numbers between the experimental groups.

A total of 30 arrays were used, each having 22960 probesets (representing genes) including controls. An expression level is associated with every gene and for each array. The 30 expression levels of a given gene form its profile. It was shown in [1] that the retinal phenotype of the KO R7E and R6/2 correlates with loss of expression for many known genes. At the same time, gain of expression was observed for other genes. Results in [1] were obtained after a 2-by-2 comparison of data in experimental groups using Wilcoxon-Mann-Whitney sum rank test, and by searches in biological databases. Here, we used FCM to cluster gene profiles and then focused our attention on some interesting cluster profiles. From the 22690 probesets in the dataset, 3020 were selected as follows. Using the CEL format files generated by the Affymetrix GeneChip Operating Software [19], we used Robust Multi-array Average (RMA) [20] for normalization and association of expression levels to probesets. Then, we selected probesets having at least one expression level greater than 6.5 in the 30 arrays. The value 6.5 corresponds to the median of all expression levels and was used as the minimum detection threshold for each probeset. A total of 13716 probesets passed out of this heuristic filter. We finally used an ANalysis Of VAriance (ANOVA) statistical test to search for differentially expressed probesets between the 6 experimental groups. The list of the final 3020 genes were obtained using a p-value threshold of 0.005 which allows to have a False Discovery Rate (FDR) of 2% [5]. Before clustering,

Table 5.1. Array numbers and their repartition between the 6 exprimental groups

#	phenotype (arrays)	mice age
1	R6/2:WT (71, 73, 79, 86)	9 weeks
2	R6/2:KO (70, 80, 82, 83)	9 weeks
3	R7E:WT (87, 88, 89, 90, 93)	3 weeks
4	R7E:KO (207, 208, 212, 213, 214)	3 weeks
5	R7E:WT (55, 56, 57, 66, 67, 68)	9 weeks
6	R7E:KO (174, 179, 180, 186, 189, 190)	9 weeks

data are standardized, i.e. transformed such that each gene has zero mean value and a standard deviation equal to one. This transformation allows performing appropriate comparison of profiles from genes which may have totally different mean expression levels.

We used Visual C++ to compute cluster membership values and centers and Matlab for generating the figures. A user friendly package of the FCM algorithm (Flora) can be found at *http://www-microarrays.u-strasbg.fr/Flora/ index.html*. However, only the case $\alpha = 2$, square of Euclidean distance, is currently available in this software. The C++ code developed for the results presented here will be added to the Flora package.

5.3.1 Results for *2-Way* Data

We first considered the data associated with the selected 3020 probesets as a *2-way* dataset. Hence, this dataset has a dimension equal to the number of arrays, $J = 30$. We began the analysis by estimating an upper bound for the fuzzy parameter for two values of α. We obtained 1.37 and 1.75 using Euclidean metric and for two values of the parameter α equal to 2.0 and 4.0, respectively. To have an estimation of the number K of clusters in the dataset, we used hierarchical agglomerative clustering, then from the dendrogram, we decided to choose $K = 30$.

We ran the FCM by setting the fuzzy parameter to $m = 1.2$. To show the advantage of using the parameter α, we ran the FCM algorithm with two values of α: 2 (square of Euclidean metric) and 4.0. The convergence threshold was set to 0.001, meaning that the algorithm stops if the Frobenius norm of the matrix corresponding to the difference between two consecutive iterations of membership value matrices is lesser than this threshold. The algorithm will also stop if the number of iterations exceeds a pre-specified maximum number of iterations (*200*). 20 runs were used with random initial solution for each and the best run result was kept. This allows the handling of the possible convergence of the FCM algorithm to a local solution. We searched the maximum membership value associated to each gene. Figure 5.1 corresponds to histograms of these

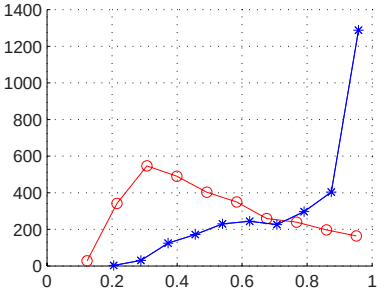


Fig. 5.1. Histograms of maxima membership values attributed to genes when using $\alpha = 2$ (circle line plot) and $\alpha = 4$ (star line plot). For the same distance metric, we increased the maxima of membership values by increasing the value of α.

maxima for the two values of α. This figure shows that we have more genes having small membership values for $\alpha = 2$ than for $\alpha = 4$.

We used a threshold level on the membership values to form clusters. We did not adopt a solution which consists of assigning each gene to a group where the membership value is the highest. Indeed, we wanted to have on the one hand, groups of genes tightly related and, on the other hand, genes having the tendency to belong to more than one group. Median values of the first maxima membership values associated to the 3020 genes are 0.4644 and 0.8735, respectively, for $\alpha = 2$ and $\alpha = 4$. We fixed the membership values threshold to 0.7. This value allows grouping 70% of the genes in dataset when α is set to 4. Note that only 22% of the 3020 genes in the dataset will be grouped if the same threshold is used with square Euclidean distance ($\alpha = 2$). The profiles of groups obtained are given in Figure 5.2.

The profile of cluster #27 shows a clear correlation with the retinal phenotype of the KO R7E and R6/2 genotypes. For this cluster, the expression level of the

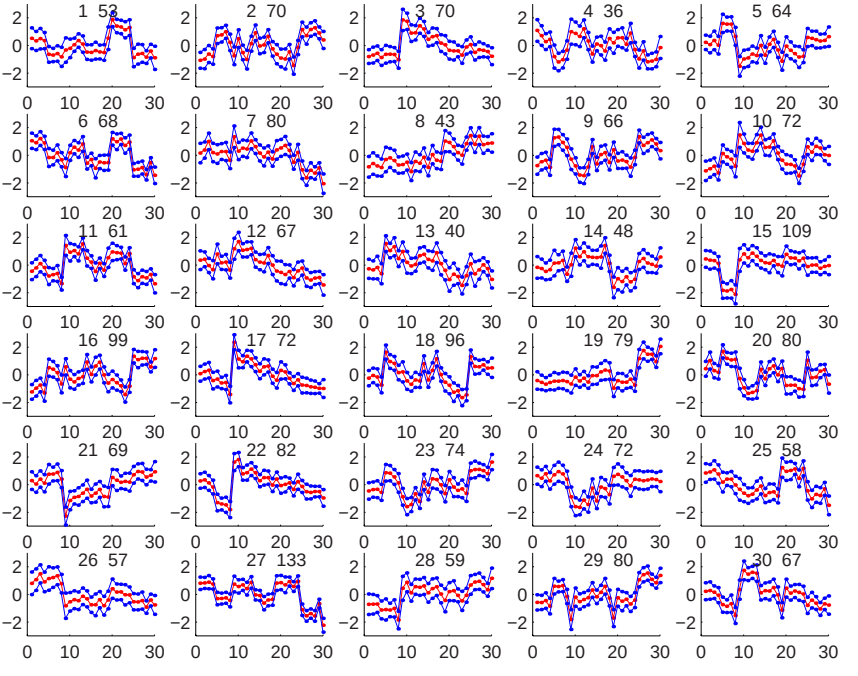


Fig. 5.2. Profiles of the 30 clusters obtained with FCM, $m = 1.2$, Euclidean distance, $\alpha = 4$, membership values threshold = 0.7. For each cluster we have 3 curves corresponding to the mean profile ± the standard deviation. Each cluster is identified by a # and its total number of genes, e.g., cluster 27 contains 133 genes. For each cluster, a point on the x-axis corresponds to an array number. Arrays are ordered in the same way as they appear in Table 5.1, i.e. 71, 73, 79, 86 70, 80, 82, 83, 87, 88, 89, 90, 93, 207, 208, 212, 213, 214, 55, 56, 57, 66, 67, 68, 174, 179, 180, 186, 189, 190).

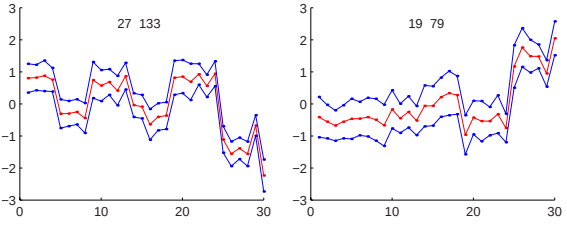


Fig. 5.3. Zoomed profiles of cluster #27 and #19 obtained with FCM and membership threshold set to 0.7

WT mice is remarkably higher than the expression level for the KO mice. Cluster #6 shows a similar profile. These two clusters contain all genes (including *Crx*, *Nrl* and *Nr2e3*) described in [1] and which are involved in the phototransduction function and morphogenesis of differentiated rod photoreceptors. Cluster #2 and #16 contain genes (including *Stat3*) having an expression level higher in the KO mice than in the WT mice, especially for mice of age 9 weeks. Cluster #19 shows, however, profile of genes which have especially higher expression level in the R7E KO genotype than in the corresponding WT mice at age 9 weeks. Figure 5.3 shows zoomed profiles for cluster #27 and #19.

5.3.2 Functional Enrichment

To verify that the link between probesets in some clusters and the Retina tissue is not a random observation, we used the web-accessible program "Database for Annotation, Visualization, and Integrated Discovery" (DAVID) [12]. DAVID allows the functional annotation of a list of genes according to shared biological information available in various databases. We first used the list of the 201 probesets in clusters 6 and 27 (see Figure 5.2) and a list of 201 probesets randomly selected from the 22690 of the Affymetrix GeneChip 430A. From the DAVID web site, *http://david.abcc.ncifcrf.gov*, we used the "Functional Annotation Tool" and uploaded our probeset lists one at a time.

The 201 probesets in the list coming from clusters 6 and 27 were associated with 172 DAVID identifiers, among them, 150 (87%) were noted as expressed in tissues. The 201 probesets randomly selected were associated with 214 DAVID identifiers. Among them, 162 (75%) were noted as expressed in tissues. We downloaded the "Tissue Expression" results in text files for local analysis which consisted of counting the number of probesets expressed in the Retina and/or the Eye tissue. We found 93 probesets (62%) and 17 probesets (10.5%), respectively, for the list coming from clusters 6 and 27 on the one hand, and the list randomly selected on the other hand. More interestingly, there were 20 probesets (13.3%), in the list coming from clusters 6 and 27, expressed only in the Retina and/or the Eye tissue. At the same time, among the 17 probesets coming from the list randomly selected, none were expressed in only the Retina and/or the Eye tissue. These results are summarized in Table 5.2. In this table, we also summarized

Table 5.2. Results obtained using DAVID Functional Annotation Tool

Probesets origin (number)	C6+C27 (201)	Random (201)	C2+C16+C19 (248)	All clusters (2124)
Expressed in tissue	150	162	187	1525
Contain {Retina, Eye}	93(62%)	17(10.5%)	39(21%)	414(27%)
Only {Retina, Eye}	20(13.3%)	0(0%)	0(0%)	42(2.7%)

results obtained using probesets coming from clusters 2, 16 and 19 on one hand, and from all clusters (membership values threshold was set to 0.7) on the other hand.

These results show that a random selection of probesets leads to a small number (10.5%) of genes expressed in the Retina/ Eye tissue. They also show that our initial filter allows the selection of probesets related (27%) to the Retina/ Eye tissue. To verify that the probesets in clusters 6 and 27 are significantly related to the Retina/ Eye tissue, we used the probability density function of the hypergeometric distribution. The computations were performed using the R environment (*see http://www.r-project.org/*). From the values in Table 5.2, the probability of having 93 probesets or more expressed in the Retina/ Eye tissue is *p = phyper(93,414,1525-414,150,lower.tail=FALSE)=5.308E-22*. This probability value is highly significant and shows that the probesets in clusters 6 and 27 are tightly related to the Retina/ Eye tissue. The p-value associated to probesets in clusters 2, 16 and 19 is *p = phyper(39,414,1525-414,187,lower.tail=FALSE)=0.9778*.

5.3.3 Results for a *3-Way* Dataset

For the results presented in this subsection, only data for the R7E model were used. In these data, two biological conditions (WT and KO) were compared at two time points (3 weeks and 9 weeks mice) leading to a *3-way* dataset. The total number of probesets (genes) is unchanged, $I = 3020$. However, we have $J_1 = 10$ and $J_2 = 12$ corresponding to the number of arrays used in the comparison with 3 weeks age mice and with 9 weeks age mice, respectively. We ran the alternating optimization FCM algorithm described in Equations (5.19)-(5.21) for a *3-way* dataset. As previously, we used the same tuning parameters but the number of cluster K was set to 12. This is because we decreased the complexity of the dataset and we wanted to highlight more relevant groups. The histogram in Figure 5.4 shows the distribution of the maxima of the membership values obtained for the 3020 genes. 2275 genes (more than 75% of genes in the dataset) have a maximum membership value greater than 0.7. Using 0.7 as a membership value threshold, we obtained the cluster profiles in Figures 5.5 and 5.6 for the 3 weeks age mice and 9 weeks age mice, respectively. Each cluster # in Figures 5.5 and 5.6 has the same number of genes, since the same membership values are

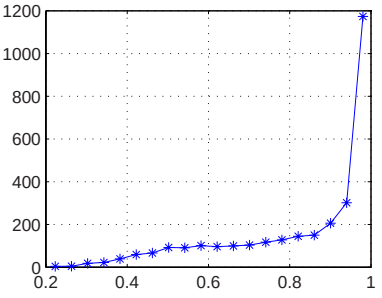


Fig. 5.4. Histogram of maxima membership values attributed to genes when using $\alpha = 4$

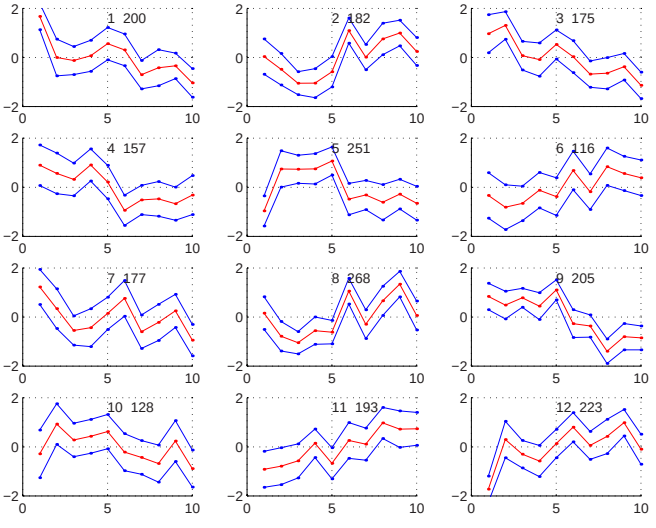


Fig. 5.5. 3 weeks age mice, profiles of the 12 clusters obtained with FCM, $m = 1.2$, Euclidean distance, $\alpha = 4$, membership values threshold = 0.7. For each cluster we have 3 curves corresponding to the mean profile ± the standard deviation. Each cluster is identified by a # and its total number of genes. The x-axis refers to an array number, i.e. 87, 88, 89, 90, 93, 207, 208, 212, 213, 214).

attributed. Differences are, however, observed for profiles associated with the 3 weeks age mice and 9 weeks age mice.

The expression level of cluster #9 is remarkably higher in the WT mice than in the KO mice for the two time points (3 weeks and 9 weeks age mice). This cluster contains the three genes (Crx, Nrl and Nr2e3) known to be related to retinal dystrophy or degenerative disorders. Many other genes (Rho, Pde6g, Txnl, Rdh12, Rs1 and Sag) related to retina disorder and not reported in [1] are present in this cluster. Cluster #5 has a profile similar to that of cluster #9.

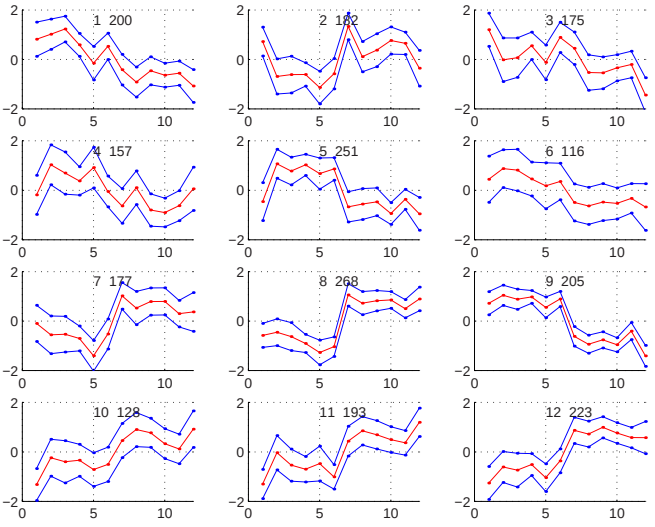


Fig. 5.6. 9 weeks age mice, profiles of the 12 clusters obtained with FCM, $m = 1.2$, Euclidean distance, $\alpha = 4$, membership values threshold = 0.7. For each cluster we have 3 curves corresponding to the mean profile ± the standard deviation. Each cluster is identified by a # and its total number of genes. The x-axis refers to an array number, i.e. 55, 56, 57, 66, 67, 68, 174, 179, 180, 186, 189, 190).

However, the first array (#55 and #87) of this cluster has, on average, a different value compared to those of the same group. We consulted the wet lab results for these arrays and found that, on average, the expression level of array #87 was lower than those of the other arrays, meaning that the RMA normalization and signal summarization seem to be inadequate for correcting certain technical problems which are encountered during data generation.

Cluster #8 has an interesting profile. In the 9 weeks age mice, its expression level is remarkably higher in the KO mice than in the WT mice. This situation is not clear for the 3 weeks age mice. Other clusters, (#12, #11, and #7) have profiles similar to that of cluster #8.

5.4 Conclusion

In this chapter, the Fuzzy C-Means clustering algorithm was used for analyzing microarray gene expression data. We outlined the concern of clustering gene expression data and then explained why we chose FCM. Then, this clustering method was presented in the context of *2-way* and *3-way* datasets. We especially discussed the choice of tuning parameters for high dimensional datasets.

We introduced a parameter for transformation of distances between all pairs of dataset features (genes) and hence allowed more contrast for detection of group

structures, if any, in high dimensional datasets. The parameter α for transforming distances can also be used for small dimensional datasets. For results, not shown here, we observed a fast convergence of the FCM algorithm when $\alpha = 4$, in comparison to the classical choice where α is set to 2.

Using some assumptions on the data distribution, we showed that the coefficient of variation (cv) of distances between all pairs of samples in a dataset is related to the data dimension. More precisely, the cv of Euclidean metric distance decreases when data dimension grows.

We used a microarray data recently published [1] to show performance of the FCM clustering method. These data were generated by the core facility Biopuces de Strasbourg (*http://www-microarrays.u-strasbg.fr*). We were able to recover clusters containing the same genes previously described. More interestingly, we also found other genes sharing the same biological function of known genes and not reported in [1]. We used a web-based program publicly available to associate functional enrichment to some genes in clusters having interesting profiles. We showed that genes in two of the clusters are primarily expressed in the Retina tissue and this cannot due to chance (small probability value).

Acknowledgments

We thank Christelle Thibault-Carpentier and Susan Y. Park for critical reading of the manuscript. This work was supported by the Centre National de la Recherche Scientifique, the Institut National de la Santé et de la Recherche Médicale and the Université Louis Pasteur de Strasbourg.

References

1. Abou-Sleymane, G., Chalmel, F., Helmlinger, D., Lardenois, A., Thibault, C., Weber, C., Mérienne, K., Mandel, J.-L., Poch, O., Devys, D., Trottier, Y.: Polyglutamine expansion causes neurodegeneration by altering the neuronal differentiation program. Hum. Mol. Genetics 15(5), 691–703 (2006)
2. 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. Proc. Natl. Acad. Sci. 96(12), 6745–6750 (1999)
3. Armstrong, S.A., Staunton, J.E., Silverman, L.B., Pieters, R., den Boer, M.L., Minden, M.D., Sallan, S.E., Lander, E.S., Golub, T.R., Korsmeyer, S.J.: MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. Nature Genetics 30, 41–47 (2002)
4. Ben-Dor, A., Shamir, R., Yakhini, Z.: Clustering gene expression patterns. J. of Comput. Biol. 6(3-4), 281–297 (1999)
5. Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: A practical and powerful approach to multiple testing. J. R. Statist. Soc. B 57, 289–300 (1995)
6. Beyer, K., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 217–235. Springer, Heidelberg (1998)

7. Bezdek, J.C.: Pattern recognition with fuzzy objective function algorithms. Plenum Press, New York (1981)
8. Bolstad, B.M., Irizarry, R.A., Astrand, M., Speed, T.P.: A comparison of normalization methods for high density oligonucleotide array data based on bias and variance. Bioinformatics 19(2), 185–193 (2003)
9. Broberg, P.: Statistical methods for ranking differentially expressed genes. Genome Biology 4(6), R 41.1–R 41.9 (2003)
10. Cromer, A., Carles, A., Millon, R., Ganguli, G., Chalmel, F., Lemaire, F., Young, J., Dembele, D., Thibault, C., le Muller, D., Poch, O., Abecassis, J., Wasylyk, B.: Identification of genes associated with tumorigenesis and metastatic potential of hypopharyngeal cancer by microarray analysis. Oncogene 23, 2484–2498 (2004)
11. Dembele, D., Kastner, P.: Fuzzy c-means method for clustering microarray data. Bioinformatics 19(8), 973–980 (2003)
12. Dennis Jr., G., Sherman, B.T., Hosack, D.A., Yang, J., Gao, W., Lane, H.C., Lempicki, R.A.: DAVID: Database for annotation, visualization, and integrated discovery. Genome Biology 4(9), R60 (2003)
13. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. 95, 14863–14868 (1998)
14. Gath, I., Geva, A.B.: Unsupervised optimal fuzzy clustering. IEEE Trans. Pattern Analysis and Machine Intelligence 11(7), 773–781 (1989)
15. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
16. Gröll, L., Jäkel, J.: A new convergence proof of fuzzy c-means. IEEE Trans. Fuzzy Systems 13(5), 717–720 (2005)
17. Gustafson, E.E., Kessel, W.C.: Fuzzy clustering with a fuzzy covariance matrix. In: Proc. of the IEEE Conference, vol. 2, pp. 761–766 (1978)
18. Höppner, F., Klawonn, F.: A contribution to convergence theory of fuzzy c-means and derivatives. IEEE Trans. Fuzzy Syst. 11(5), 682–694 (2003)
19. Affymetrix cgos, free technical support software, `http://www.affymetrix.com/support/index.affx`
20. Irizarry, R.A., Bolstad, B.M., Collin, F., Cope, L.M., Hobbs, B., Speed, T.P.: Summaries of affymetrix genechip probe level data. Nucleic Acids Research 31(4), e15 (2003)
21. Iyer, V.R., Eisen, M.B., Ross, D.T., Schuler, G., Moore, T., Lee, J.C.F., Trent, J.M., Staudt, L.M., Hudson Jr., J., Bogoski, M.S., Lashkari, D., Shalon, D., Botstein, D., Brown, P.O.: The transcriptional program in the response of human fibroblasts to serum. Science 283, 83–87 (1999)
22. Krishnapuram, R., Kim, J.: Clustering algorithms based on volume criteria. IEEE Trans. Fuzzy Systems 8(2), 228–236 (2000)
23. Milligan, G.M., Cooper, M.C.: An Examination of Procedures for Determining the Number of Clusters in a Data Set. Psychometrika 50(2), 159–179 (1985)
24. Morgan, B.J.T., Ray, A.P.G.: Non-uniqueness and inversions in cluster analysis. Appl. Statist. 44, 117–134 (1995)
25. Sato, M., Sato, Y., Jain, L.C.: Fuzzy clustering models and applications. Physica-Verlag (1997)
26. Sharan, R., Shamir, R.: A clustering algorithm with application to gene expression analysis. In: Proc. AAAI - ISMB, CLICK 2000, pp. 307–316 (2000)

27. Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., Merisov, J.P.: Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc. Natl. Acad. Sci. USA 102, 15545–15550 (2005)
28. Tamayo, P., Slonim, D., Mesirov, J.P., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S., Golub, T.R.: Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. Proc. Natl. Acad. Sci. USA 96, 2907–2912 (1999)
29. Tavazoie, S., Hughes, J.D., Campbell, M.J., Raymond, I., Cho, R.I., Church, G.M.: Systematic determination of genetic network architecture. Nature Genetics 22, 281–285 (1999)
30. Theodorisis, S., Kouthroumbas, K.: Pattern recognition. Academic Press, New-York (1999)
31. Tian, L., Greenberg, S.A., Kong, S.W., Altschuler, J., Kohane, I.S., Park, P.J.: Discovering statistically significant pathways in expression profiling studies. Proc. Natl. Acad. Sci. USA 102(38), 13544–13549 (2005)
32. Tseng, G.C., Oh, M.-K., Rohlin, L., Liao, J.C., Wong, W.H.: Issues in cdna microarray analysis: quality filtering, channel normalization, models of variations and assessment of gene effects. Nucleic. Acids Res. 29(12), 2549–2557 (2001)
33. Tusher, V.G., Tibshirani, R., Chu, G.: Significance analysis of microarrays applied to the ionizing radiation response. Proc. Natl. Acad. Sci. USA 98(9), 5116–5121 (2001)
34. Wicker, N., Dembele, D., Raffelsberger, W., Poch, O.: Density of points clustering, application to transcriptomic data analysis. Nucleic Acids Research 30(18), 3992–4000 (2002)

6

Fuzzy Patterns and GCS Networks to Clustering Gene Expression Data

Daniel Glez-Peña[1], Fernando Díaz[2], Florentino Fdez-Riverola[1], José R. Méndez[1], and Juan M. Corchado[3]

[1] Department of Computer Science, University of Vigo, Ourense, 32004, Spain
{dgpena,riverola,moncho.mendez}@uvigo.es
[2] Department of Computer Science, University of Valladolid, Segovia, 40005, Spain
fdiaz@infor.uva.es
[3] Department of Computer Science, University of Salamanca, Salamanca, 37008, Spain
corchado@usal.es

Summary. The advent of DNA microarray technology has supplied a large volume of data to many fields like machine learning and data mining. Gene expression profiles are composed of thousands of genes at the same time, representing complex relationships between them. In this context, intelligent support is essential for managing and interpreting this great amount of information. One of the well-known constraints specifically related to microarray data is the large number of genes in comparison with the small number of available experiments. In this situation, the ability of design methods capable of overcoming current limitations of state-of-the-art algorithms is crucial to the development of successful applications. In this chapter we present a flexible framework for the task of feature selection and classification of microarray data. Dimensionality reduction is achieved by the application of a supervised fuzzy pattern algorithm able to reduce and discretize existing gene expression profiles. An informed growing cell structures network is proposed for clustering biological homogeneous experiments starting from the previous simplified microarray data. Experimental results over different data sets containing acute myeloid leukemia profiles show the effectiveness of the proposed method.

6.1 Introduction

The advent of microarray technology has become a fundamental tool in genomic research, making it possible to investigate global gene expression in all aspects of human disease. In particular, cancer genetics based on the analysis of cancer genotypes, provides a valuable alternative to cancer diagnosis in both theory and practice [14]. In recent years, there has been an explosion of methods that analyze gene expression arrays to produce long lists of genes that express differentially in distinct cellular states. Gene expression arrays provide a great amount of valuable biological information, although it represents only a suspicion about the processes taking place within the whole cell.

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 103–125.
springerlink.com

Recent studies in human cancer have demonstrated that microarrays can be used to develop a new taxonomy of cancer, including major insights into the genesis, progression, prognosis, and response to therapy on the basis of gene expression profiles [26]. The automatic classification of cancer patients has been a promising approach in cancer diagnosis since the early detection and treatment can substantially improve the survival rates. For this task, a number of successful machine learning approaches have been proposed in the literature, including support vector machines (SVM) [4], artificial neural networks (ANN) [34, 21], k-nearest neighbor (k-nn) [25] and hierarchical clustering [1] methods, among others.

Since the number of examined genes in an experiment runs to the thousands, a major problem with the application of existing clustering and classification techniques is the huge number of attributes (genes) in the existing datasets. Gene reduction in microarray data is extremely important because: (i) it generally reduces the computational cost of machine learning techniques, (ii) it usually increases the accuracy of classification algorithms [6] and (iii) it provides clues to researches about genes that are important in a given context (i.e. biomarkers for certain diseases).

In this context, several methods derived from machine learning and multivariate statistical analyses have been applied to gene selection/dimension reduction in the field of microarray data. On the one hand, there are works on applying genetic algorithms [23, 7], wrapper approaches [2], support vector machines [15, 5] or spectral biclustering [24] to achieve significant reduction rates. On the other hand, the utilization of partial least squares (PLS), sliced inverse regression (SIR) or principal component analysis (PCA) have been shown highly useful for classification with gene expression data [6]. Other approaches focus their attention on redundancy reduction and feature extraction [19, 28], as well as the identification of similar gene classes making prototypes-genes [17]. One way or another, the selected method has to pursue two main goals: (i) reduce the cost and complexity of the classifier and (ii) improve the accuracy of the model.

In recent years, the number and variety of applications of fuzzy logic have increased significantly. The applications range from consumer products such as cameras, camcorders, washing machines, and microwave ovens to industrial process control, medical instrumentation, and decision-support systems. But, what is meant by fuzzy logic?. In a narrow sense, fuzzy logic represents a logical system, which is an extension of multi-valued logic. However, from a broader perspective fuzzy logic is almost synonymous with the theory of fuzzy sets, a theory which relates to classes of objects with unsharp boundaries in which membership is a matter of degree [31, 10]. Another basic concept in fuzzy logic, which plays a central role in most of its applications, is that of a fuzzy if-then rule or, simply, fuzzy rule. Although rule-based systems have a long history of use in AI, what is missing in such systems is a mechanism for dealing with fuzzy consequents and fuzzy antecedents. In fuzzy logic, this mechanism is provided by the calculus of fuzzy rules.

In this context, a trend that is growing in visibility relates to the use of fuzzy logic in combination with neurocomputing and genetic algorithms. More generally, fuzzy logic, neurocomputing, and genetic algorithms may be viewed as the principal constituents of what might be called soft computing [32]. Unlike the traditional hard computing, soft computing accommodates the imprecision of the real world. In this sense, the guiding principle of soft computing is to exploit the tolerance for imprecision, uncertainty, and partial truth to achieve tractability, robustness, and low solution cost.

In this work, a gene fuzzy discretization process is proposed prior to the application of a supervised discriminant fuzzy pattern generation algorithm. The similitude between existing fuzzy labels (representing genes) is later used for defining a metric that will be applied for boosting cell distance computation in our GCS (*Growing Cell Structures*) network. The combination of biological knowledge (drawn from the gene selection process) into the operation of the neural network powers its self explanatory capability. Results over two different microarray datasets are reported showing the feasibility of the proposed method combination.

The rest of the chapter is organized as follows. The next section describes the proposed fuzzy pattern algorithm for reducing and discretizing gene expression data. Then, we present and explain in detail the improved GCS neural network. This lays the groundwork for introducing our GENECBR tool for multiple-microarray analysis in which the proposed techniques were implemented. Later, we present the case study and analyze the evaluation results. The chapter concludes with a discussion of the main themes.

6.2 Filtering Superfluous Genes

Classical reduction dimension methods applied to microarray data tend to identify differentially expressed genes from a set of microarray experiments [33]. A differentially expressed gene is a gene which has the same expression level for all examples of the same class, but different for those examples belonging to different classes. The relevance value of a gene depends on its capacity of being differentially expressed. However, a non-differentially expressed gene will be considered irrelevant and will be removed from the classification process even though it might well contain information that would improve the classification accuracy. For this reason, the task addressed here is slightly different from that of feature selection for gene expression based classifiers [18, 30].

The *fuzzy set* concept is crucial to perform our analysis in terms of knowledge representation, since this notion is used to define the adjectives that describe the different expression levels of a gene given by the raw microarray data. We are not interested in the construction of a fuzzy inference model, where the system provides a mapping from a given input to an output using fuzzy logic in terms of a list of fuzzy rules (if-then statements) [20, 16, 29]. Instead of this, we are interested in the representation of expression levels of a gene as a vague concept, where it can be admitted the possibility of partial membership in it.

The fuzzy set notion is intimately related to the fuzzy logic, where the truth of any statement becomes a matter of degree. The major advantage that fuzzy logic offers is the ability to reply to a yes-no question with a not-quite-yes-or-not answer. Humans do this kind of thing all the time, but it is a rather new trick for computers. The way in which each point of the input space is mapped to a degree of membership (between 0 and 1) is given by a membership function.

The function itself can be an arbitrary curve whose shape can be defined as a function that is suitable from the point of view of simplicity, convenience, speed, and efficiency. In this work, two types of membership functions are used. Firstly, a polynomial approximation of a Gaussian membership function which achieve smoothness for the degree of membership of 'normal' expression levels of a gene, and secondly, a polynomial approximation of two sigmoidal membership functions which are able to specify asymmetric membership functions for the 'low' and 'high' expression levels.

The whole algorithm comprises of three main steps. First, we represent each gene value in terms of one from the following linguistic labels: LOW, MEDIUM, HIGH and their intersections LOWMEDIUM and MEDIUMHIGH. The output is a *fuzzy microarray descriptor* (FMD) for each existing sample (microarray). The second phase aims to find all genes that best explain each class, constructing a supervised *fuzzy pattern* (FP) for each pathology. Starting from the previous obtained FPs, our proposed method is able to discriminate those genes that can provide a substantial discernibility between existing classes, generating a unique *discriminant fuzzy pattern* (DFP).

6.2.1 Discretizing Microarray Data Using Fuzzy Labels

Given a set of n expressed sequence tags (ESTs) or genes, the discretization process is based on determining the membership function of each gene to the three linguistic labels LOW, MEDIUM and HIGH. These membership functions are defined by a polynomial function that approximates a Gaussian membership function, where its center and amplitude depend on the mean and on the variability of the available data, respectively (see Figure 6.1). The MEDIUM membership function is considered symmetric whereas the LOW and HIGH functions are asymmetric in the extremes (see [11] for more details about the mathematical background).

Our method defines a threshold value, θ, which need to be established in order to discretize the original data in a binary way. For concrete values of threshold θ, specific zones of the gene values domain for which none of the labels will be activated can exist (neighbor region of the intersection of labels MEDIUM and HIGH in Figure 6.1).

This fact must be interpreted as the specific value of the feature is not enough to assign it a significant linguistic label at the significance degree of membership fixed by threshold θ. On the other hand, one value can simultaneously activate two linguistic labels, since at the significance level given by θ, any assignment of the measure to a linguistic label is significant (neighbor region of the intersection of labels LOW and MEDIUM in Figure 6.1).

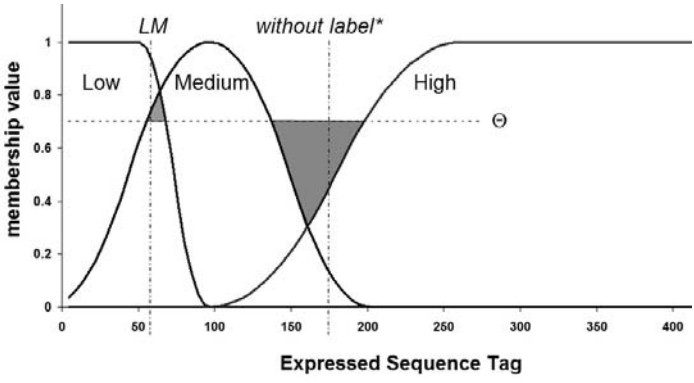


Fig. 6.1. Shape of membership functions for a specific gene and possible assigned labels given a threshold $\theta = 0.7$

6.2.2 Assembling a Supervised Fuzzy Pattern of Representative Genes

A *fuzzy pattern* is a higher concept built from a set of FMDs belonging to the same class, and it can be viewed as a prototype of them. The FP corresponding to a given class is constructed selecting the genes with a label which has a relative frequency of appearance equal to or greater than a predefined ratio π ($0 < \pi \leq 1$). Therefore, the FP captures relevant and common information about the discretized gene expression levels of the FMDs that summarizes.

The predefined ratio π controls the degree of exigency for selecting a gene as a member of the pattern, since the higher the value of π, the fewer the number of genes which make up the FP. The pattern's quality of fuzziness is given by the fact that the labels, which make it up, come from the linguistic labels defined during the transformation into FMD of an initial observation. Moreover, if a specific label of a gene is very common in all the examples belonging to a given class, this feature will be selected to be included in the FP. Therefore, a frequency-based criterion is used for selecting a gene as part of the fuzzy pattern.

6.2.3 Recognizing Valuable Genes

The goal of gene selection is to determine a reduced set of genes, which are useful to classify new samples given the existing knowledge. Now, we are interested in those genes that allow us to discriminate a given class with regard to the others.

Here, we introduce the notion of *discriminant fuzzy pattern* with regard to a collection of FPs. A DFP version of a FP only includes those genes that can serve to differentiate it from the rest of the patterns. Therefore, the computed DFP for a specific FP is different depending on what other FPs are compared with it. It's not surprising that the genes used to discern a specific class from others (by mean of its DFP) will be different if the set of rival classes also changes. The

pseudo code algorithm used to compute the final DFP containing the selected genes can be consulted on [9].

6.3 Clustering Biologically Homogeneous Gene Expression Data

In this work we propose the use of our DFP filter as a gene selection strategy to generate the input data for a GCS neural network. The target goal of the GCS network is to group together those samples (microarrays) that are most similar, but only taking into account the genetic information provided by the previously selected genes.

6.3.1 Growing Cell Structures Networks

GCS neural networks [12] constitute an extension to Kohonen's self-organizing maps [22], and are only one member in the family of self-organizing incremental models. GCS networks have the advantage of being able to automatically construct the network topology, and to support easy visualization of semantic similarity in high-dimensional data. More importantly, the extracted knowledge that is relevant to clustering can provide meaningful explanations for the clustering process and useful insight into the underlying domain.

It is important to highlight that the final goal of our GCS network is to cluster all patients that are genetically similar given a filtered and discretized group of genes (FMD_{DFP}), and without taking into account their previous assigned classes. Our proposed method aims to find new relations between the patients even now unknown. Therefore, it is possible and not contradictory to group together patients suffering different (but genetically related) diseases. Such a topology has the added advantage that inter-cluster distances can be precisely quantified. Since such networks contain explicit distance information, they can be used effectively to (*i*) represent an indexing structure which indexes sets of related patients and to (*ii*) serve as a similarity measurement between individual patients.

To illustrate the working model of the GCS network used in our framework, a two-dimensional space is used, where the cells (neurons) are connected and organized into triangles [13]. Each cell in the network is associated with a weight vector, w, of the same dimension as the previous selected group of genes (FMD_{DFP}). At the beginning of the learning process, the weight vector of each cell is initialized with random values [13]. The basic learning process in a GCS network consists of topology modification and weight vector adaptations carried out in three steps.

In the first step of each learning cycle, the cell c, with the smallest distance between its weight vector, w_c, and the actual FMD_{DFP} is chosen as the *winner cell* or best-match cell. The selection process is succinctly defined by using the Euclidean distance measure as indicated in Expression (6.1) where O denotes the set of cells within the structure at a given point in time.

$$c : \|FMD_{DFP} - w_c\| \leq \|FMD_{DFP} - w_i\| \ ; \ \forall i \in O \tag{6.1}$$

The second step of the learning process consists of the adaptation of the weight vector, w_c, of the winning cell, and the weight vectors, w_n, of its directly connected neighboring cells, N_c, by means of Equations (6.2) and (6.3).

$$w_c(t+1) = w_c(t) + \varepsilon_c(FMD_{DFP} - w_c) \tag{6.2}$$

$$w_n(t+1) = w_n(t) + \varepsilon_n(FMD_{DFP} - w_n); \forall n \in N_c, \tag{6.3}$$

where ε_c and ε_n represent the learning rates for the winner and its neighbors respectively, belonging to the [0, 1] interval, and N_c stands for the set of direct neighbor cells of the winning cell, c.

In the third step, a *signal counter*, τ, is assigned to each cell, which reflects how often a cell has been chosen as winner. Equations (6.4) and (6.5) define how the signal counter is updated with parameter α acting as a constant rate of counter reduction for the rest of the cells at the current learning cycle, t.

$$\tau_c(t+1) = \tau_c(t) + 1 \tag{6.4}$$

$$\tau_i(t+1) = \tau_i(t) - \alpha\,\tau_i(t);\ i \neq c \tag{6.5}$$

Growing cell structures also modify the overall network structure by inserting new cells into those regions that represent large portions of the input data (genetically similar patients), or removing cells that do not contribute to the input data representation. The cell deletion policy has not been used in our work due to the lack of great amounts of data. The adaptation process is then performed after a fixed number of learning cycles of input presentations (epochs). Therefore, the overall structure of a GCS network is modified through the learning process by performing only cell insertion. Equations (6.6), (6.7) and (6.8) define the rules that govern the insertion behavior of the network.

$$h_i = \tau_i / \sum\nolimits_j \tau_j\ ;\ \forall i, j \in O \tag{6.6}$$

$$q : h_q \geq h_i\ ;\ \forall i \in O \tag{6.7}$$

$$r : \|w_r - w_q\| \geq \|w_p - w_q\|\ ;\ \forall p \in N_q. \tag{6.8}$$

Insertion starts with selecting the cell, which served the most often as the winner, on the basis of the signal counter, τ. The cell, q, with the highest relative counter value, h, is selected. The neighboring cell, r, of q with the most dissimilar weight vector is determined using Expression (6.8). In this expression, N_q denotes the set of neighboring cells of q. A new cell, s, is inserted between the cells q and r, and the initial weight vector, w_s, of this new cell is set to the mean of the two existing weight vectors, w_q and w_r. Finally, the signal counters, τ, in the neighborhood, N_s, of the newly inserted cell, s, are adjusted. The new signal counter values represent an approximation to a hypothetical situation where s would have been existed since the beginning of the process.

6.3.2 Boosting Cell Distance Computation

An important issue in the GCS network operation is the distance calculation between two cells representing two different classes in the gene expression domain. In this work, we propose to hybridize our GCS network by using biological knowledge for distance computation. Every time the network needs to compute the distance between two nodes or one node and the actual input pattern FMD_{DFP}, Equation 6.1 is used.

Given that each position in the weight vector w_c stores a gene expression value for a selected gene, we can use the similarity between linguistic labels as a measure of the relation between each position in w_c and the corresponding value in the FMD_{DFP} pattern, since we assume that

$$\|\mathbf{X} - \mathbf{Y}\| \propto \frac{1}{\sum sim(X_i, Y_i)}$$

From conventional set theory, the similarity among two sets can be computed by the ratio between the cardinal of the intersection of the two sets and the cardinal of the reference set. In this case, it has been considered that the fuzzy intersection of two fuzzy sets A and B (represented by its membership functions μ_A and μ_B, respectively) is given by the application of the min operator to the

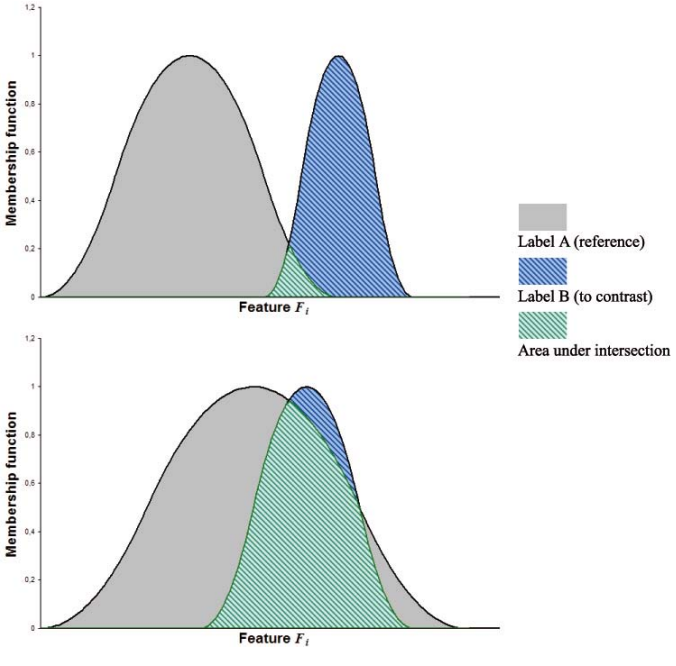


Fig. 6.2. Interpretation of similarity among fuzzy sets

two membership functions, namely, $\mu_{A\cap B} = \min\{\mu_A, \mu_B\}$. Since the cardinality operator can be replaced by the integral operator, the metric $sim(A, B)$ can be computed by Equation (6.9), where similarity ranges from 0 (total dissimilarity) to 1 (total similarity). A graphical interpretation of the similarity among fuzzy sets can be viewed in Figure 6.2.

$$sim(A, B) = \frac{|A \cap B|}{|A|} = \frac{\int_{x_1}^{x_2} \min\{\mu_A(x), \mu_B(x)\}\, dx}{\int_{x_1}^{x_2} \mu_A(x) dx} \tag{6.9}$$

In Equation (6.9) the area representing the intersection of two fuzzy sets ($\int_{x_1}^{x_2} \min\{\mu_A(x), \mu_B(x)\}\, dx$) is calculated by parts. In the general case, it is necessary to take into consideration four different zones as depicted in Figure 6.3, where c_L, c_M stand for the centers of membership functions LOW and MEDIUM respectively, and λ_L, λ_M represent the amplitude of these functions. The join point x_{cut} is calculated by means of Equation (6.10).

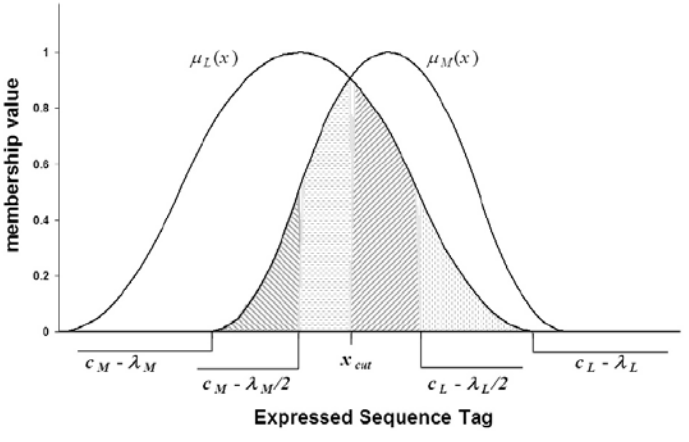


Fig. 6.3. Intersection of linguistic labels LOW and MEDIUM without adjustment

$$x_{cut} = \frac{c_L\lambda_M + c_M\lambda_L}{\lambda_L\lambda_M} \tag{6.10}$$

At this point, the analytical calculation of the integrals must be made. After some calculus, facilitated by the fact that the defined membership functions are polynomial, a closed form for these integrals is obtained. As it is shown in Figure 6.3 the total area under the curve can be split into two sections, and therefore, the similarity among LOW and MEDIUM labels is given by Expression (6.11):

$$sim(L,M) = \frac{\int_{c_M-\lambda_M}^{c_L+\lambda_L} \min(\mu_L(x),\mu_M(x))dx}{\int_{c_L-\lambda_L}^{c_L+\lambda_L} \mu_L(x)dx} = \frac{\int_{x_{\min}}^{x_{cut}} \min(\mu_L(x),\mu_M(x))dx + \int_{x_{cut}}^{c_L+\lambda_L} \min(\mu_L(x),\mu_M(x))dx}{\int_{c_L-\lambda_L}^{c_L+\lambda_L} \mu_L(x)dx} \quad (6.11)$$

where x_{cut} is given by Expression (6.10) and $x_{\min} = \max\{c_L - \lambda_L, c_M - \lambda_M\}$.

As it is mentioned previously, the mathematical definitions of membership functions $\mu_L(x), \mu_M(x)$,and $\mu_H(x)$can be found in [11], but they are not included here due space limitations. The following results, Expressions (6.12) to (6.15), are derived from these membership functions. Firstly, the area under the $\mu_L(x)$ membership function can be computed directly, as shown below:

$$\int_{c_L-\lambda_L}^{c_L+\lambda_L} \mu_L(x)dx = \int_{c_L-\lambda_L}^{c_L} dx + \int_{c_L}^{c_L+\lambda_L/2} \mu_L(x)dx + \int_{c_L+\lambda_L/2}^{c_L+\lambda_L} \mu_L(x)dx = \lambda_L + \tfrac{5}{12}\lambda_L + \tfrac{1}{12}\lambda_L = \tfrac{3}{2}\lambda_L. \quad (6.12)$$

In the same way, the area under the $\mu_M(x)$ membership function can be also computed efficiently:

$$\int_{c_M-\lambda_M}^{c_M+\lambda_M} \mu_M(x)dx = 2\left\{\int_{c_M}^{c_M+\lambda_M/2} \mu_M(x)dx + \int_{c_M+\lambda_M/2}^{c_M+\lambda_M} \mu_M(x)dx\right\} = 2\left\{\tfrac{5}{12}\lambda_M + \tfrac{1}{12}\lambda_M\right\} = \lambda_M. \quad (6.13)$$

Finally, and assuming that any expression level in the range$c - \lambda < x < c + \lambda$, where the membership degree is greater than 0, can be expressed in parametric form by $x = c + t \cdot \lambda$ with $t \in [-1, 1]$, the integral below the curve branch of each one of the considered membership functions is given by Expression (6.14).

$$\int_c^{c+t\lambda} \mu(x)dx = \int_{c-t\lambda}^{c} \mu(x)dx = \begin{cases} t\left(1 - \frac{2}{3}t^2\right)\lambda, & \text{if } 0 \le t \le \frac{1}{2} \\ \frac{5}{12}\lambda + \frac{2}{3}\left(t - \frac{1}{2}\right)\left(t^2 - \frac{5}{2}t + \frac{7}{4}\right)\lambda, & \text{if } \frac{1}{2} < t < 1 \\ \frac{1}{2}\lambda, & \text{if } t \ge 1 \end{cases} \quad (6.14)$$

Therefore, the similarity among LOW and MEDIUM labels given by Equation (6.11) can be rewritten as follows:

$$sim(L,M) = \left\{\tfrac{1}{2}\lambda_L - \int_{c_L}^{c_L+t_{cut}\lambda_L} \mu_L(x)dx\right\} + \left\{\int_{c_M-t_{\min}\lambda_M}^{c_M} \mu_M(x)dx - \int_{c_M-t_{cut}\lambda_M}^{c_M} \mu_M(x)dx\right\} / \tfrac{3}{2}\lambda_L, \quad (6.15)$$

where $t_{cut} = (c_M - x_{cut})/\lambda_M$and $t_{\min} = (c_M - x_{\min})/\lambda_M$. Solving the partial integrals in this expression with the result given in Equation (6.14), the entire integral can be computed in closed form.

The values of similarity among all membership functions belonging to each gene are precomputed at the same time that the parameters of these functions (centers and amplitudes) are determined.

6.4 Integrating Fuzzy Patters and GCS Networks Inside geneCBR

In light of the fast growth in DNA technology there is a compelling demand for tools able to perform efficient, exhaustive and integrative analyses of multiple microarray datasets. Specifically, what is particularly evident is the need to link the results obtained from these new tools with the wealth of clinical information. The final goal is to bridge the gap existing between biomedical researchers and pathologists or oncologists providing them with a common framework of interaction. To overcome such difficulty we have developed geneCBR [8], a freely available software tool that allows the use of combined techniques that can be applied to gene selection, clustering, knowledge extraction and prediction[1]. In *diagnostic mode*, geneCBR employs a case-based reasoning model that incorporates a set of fuzzy prototypes for the retrieval of relevant genes, a growing cell structure network for the clustering of similar patients and a proportional weighted voting algorithm to provide an accurate diagnosis.

geneCBR was implemented to support integrative work for inter-disciplinary research groups working together in order to design, implement and test new techniques for supervised and unsupervised cancer classification and clustering. Figure 6.4 (left) shows this user-dependent architecture:

- *Pathologists or oncologists*: geneCBR (*diagnostic mode*) implements an effective and reliable system able to diagnose cancer subtypes based on the analysis of microarray data using a CBR architecture (see Figure 6.4(right)).
- *Biomedical researches*: geneCBR (*expert mode*) offers a core workbench for designing and testing new techniques and experiments.
- *Programmers*: geneCBR (*programming mode*) includes an advanced edition module for run-time modification of previous coded techniques based on BeanShell[2].

geneCBR is written entirely in Java 1.5 and portable across multiple operating systems and platforms. It is simple to install and easy to update through the utilization of Java Web Start technology[3] which ensures the execution of the last available version.

[1] http://www.genecbr.org/
[2] http://www.beanshell.org/
[3] http://java.sun.com/products/javawebstart/

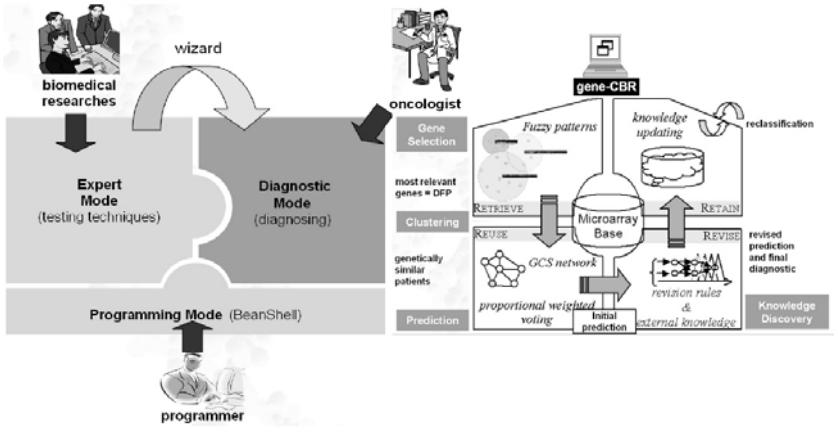


Fig. 6.4. (left) Logic architecture and interfaces of geneCBR system. (right) Life-cycle of geneCBR system working in *diagnostic mode*.

6.4.1 Core Functions and Features

geneCBR *expert mode* can load microarray expression datasets from any platform as input, as long as the data coming from different experiments have been summarized into a matrix of normalized expression values (microarray base). The input file should contain text fields with comma as separator and can hold lines starting with the number sign ('#') for commenting meta-data information. geneCBR includes (*i*) a co-expression analysis module, (*ii*) a discriminant expression analysis module, (*iii*) a supervised/unsupervised classification module and (*iv*) various *add-ins* such us the net explorer module.

The *co-expression analysis module* is used to identify sets of genes simultaneously co-expressed in multiple datasets belonging to patients suffering the same kind of cancer. Briefly, given a set of microarrays which are well classified, for each class a *fuzzy pattern* (FP) can be constructed from the fuzzy microarray descriptor (FMD) associated with each one of the microarrays (see Section 6.2.2). The FMD is a comprehensible description for each gene in terms of a linguistic label. Figure 6.5 shows how these membership functions are calculated by geneCBR.

The *discriminant expression analysis module* is developed to derive sets of genes expressed differentially in several FPs. The objective is to select those genes that allow us to discriminate a new microarray from one class with regard to the others. Here we introduce the notion of discriminant fuzzy pattern (DFP) with regard to a collection of fuzzy patterns (see Section 6.2.3). A DFP version of a FP only includes those genes that can serve to differentiate it from the rest of the patterns. A consequence is that the computed DFP for a specific FP is different depending on which other FPs are compared with it. It's not surprising that the genes used to discern a specific class from others (by mean of its DFP) will be different if the set of rival classes also changes.

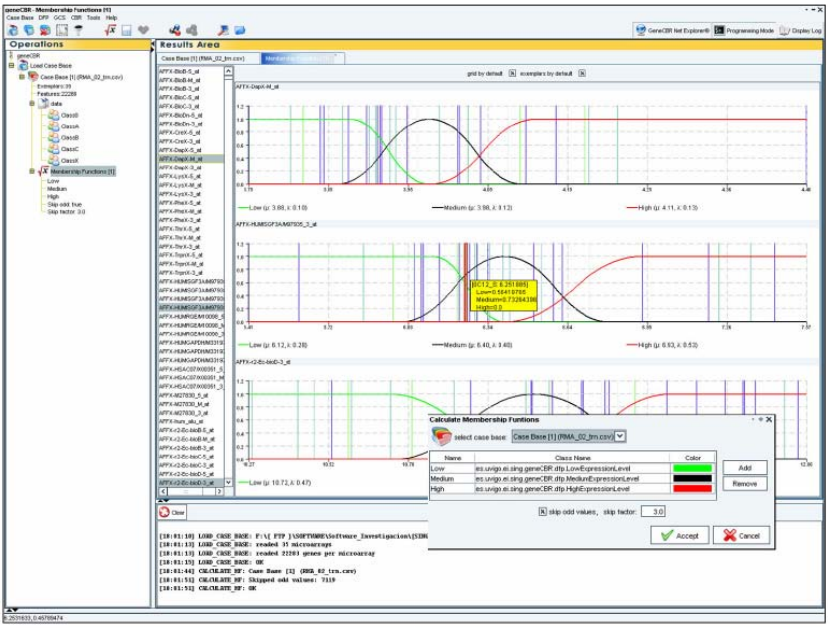


Fig. 6.5. Fuzzy membership functions representing each gene in the microarray data base

In the *supervised and unsupervised clustering module* the DFP derived from the discriminant expression analysis module serves as a filter to select a reduced number of relevant and representative genes, allowing other artificial intelligence techniques to be able to tackle the high-dimensional data. GENECBR incorporates a GCS neural network able to cluster all patients that are genetically similar given a selected group of genes and without taking into account their previous assigned classes (see Section 6.3.1). Figure 6.6 shows a diagram of the whole algorithm including the fuzzy systems and the GCS network.

The practice of biomedical research seeks to comprehend the complexity of complex organisms, or their subsystems, by combining many different kinds of data to improve existing knowledge. In current practice, as experts explore their data, they typically create manual, *ad hoc* connections among software tools and databases, cutting and pasting queries, creating temporary files, running web searches and taking notes. GENECBR includes an Internet explorer module able to gather additional information (gene annotations, public gene ids, biological functions, relevant related articles from PubMed/MedLine, etc.) in order to facilitate the integration of several sources of information. Figure 6.7 shows a typical screenshot of the query toolbox integrated within the system. GENECBR always keeps their local databases updated, downloading new information as soon as it is available in Internet.

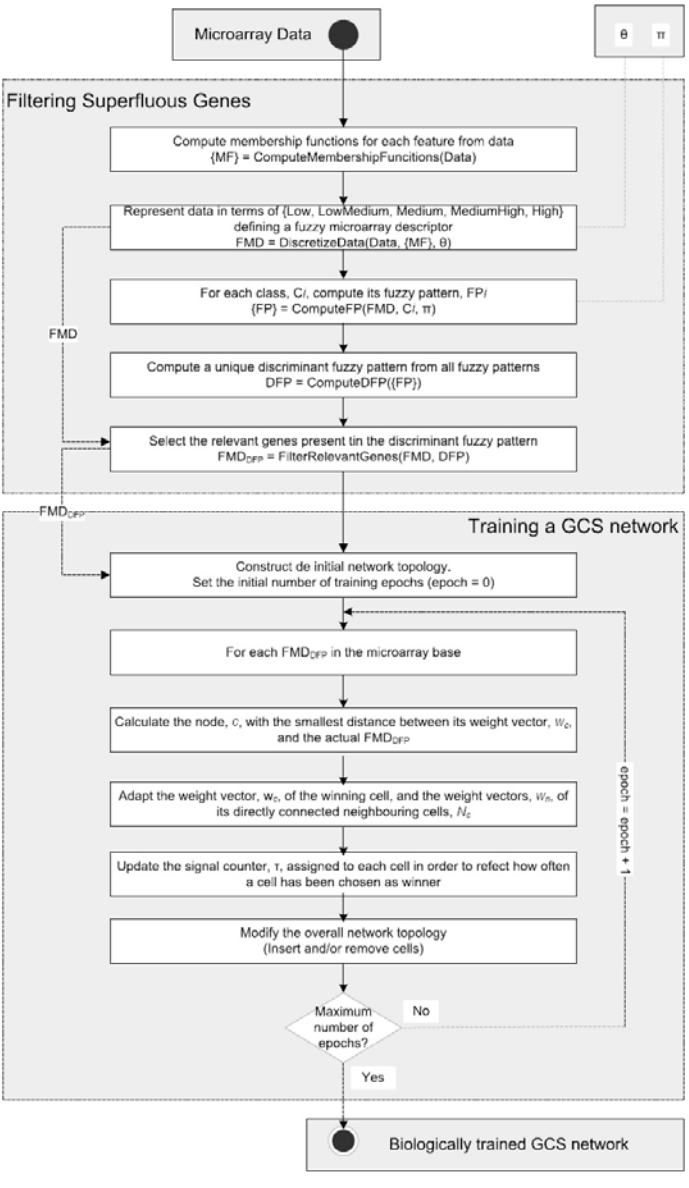


Fig. 6.6. Integrated flowchart of the proposed algorithm

Every time the biomedical research group finishes their work GENECBR provides a guided 4-step wizard to setup GENECBR *in diagnostic mode* (see Figure 6.8). With this configuration, the application is ready to receive a new microarray experiment and perform all the programmed tests in only one step.

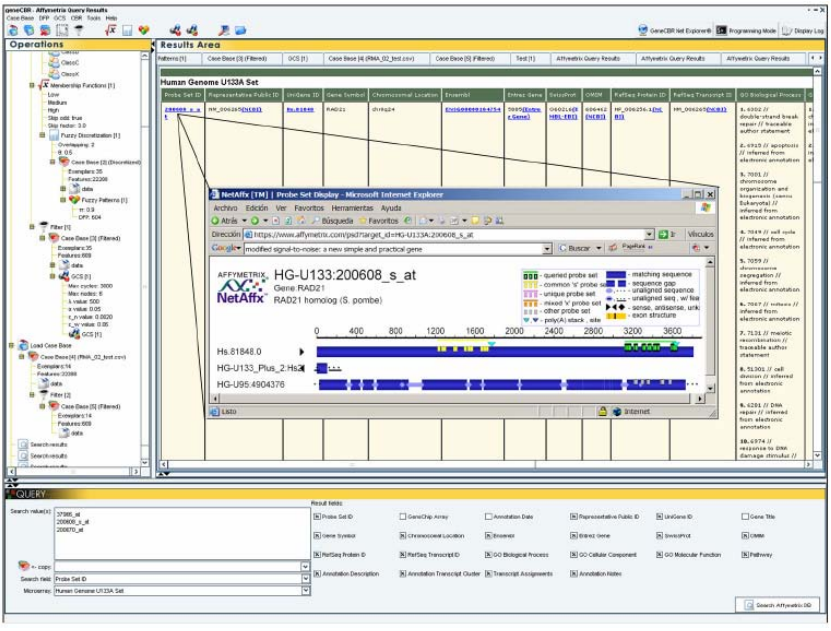


Fig. 6.7. Query results and add-ins available in geneCBR

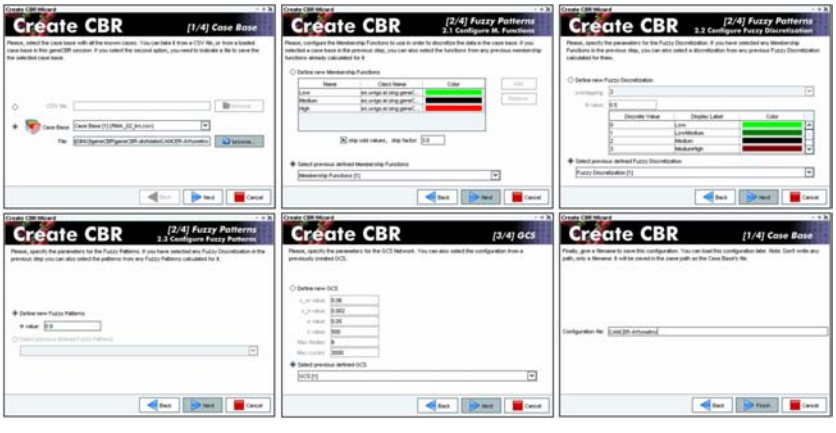


Fig. 6.8. geneCBR wizard for configuring *diagnostic mode*

6.5 Case Study and Evaluation

Acute myeloid leukemia (AML) is a heterogeneous group of hematological cancers with marked differences in their response to chemotherapy. As in many other human cancers, the diagnosis and classification of AML have been based

on morphological, cytochemical and immunophenotipic features. More recently, genetic features have helped to define biologically homogeneous entities within AML. Karyotype is the most important independent prognostic factor and therefore the most useful parameter for stratifying patients into risk groups. Thus, the favorable outcome group is composed of well-defined subtypes in terms of cytogenetics: t(15;17), inv(16) and t(8;21).

In contrast, the correlation between morphologic characteristics, genetic abnormalities and prognostic features is more inconsistent in the remaining AML. In this case, there are a significant number of AML patients (above the 50% of the whole) which are not yet classified into well-characterized subtypes of AML. In this context, the analysis of gene expression profiles of tumors using microarray technology has become a powerful tool for classifying hematopoietic neoplasms [14].

The goal of this study is to test the performance of the proposed technique in order to classify correctly patients which suffer from a well-known type of AML and which come from two different studies.

6.5.1 Gene Expression Data Sets

This section describes the available data which have been used in our experimentation. For this purpose we have employed two different data sets.

The first data set (herein, referred to as *EUMC-Rotterdam*) contains the gene expression profiles in samples of peripheral blood of bone marrow from 285 patients with AML using Affymetrix U133A gene chips. This data set was used by [27] in a study about the identification of useful gene expression profiles in AML from a prognostic Al point of view. The gene expression profiles of 8 healthy persons are also available in this data set. The data comes from the Department of Hematology, Erasmus University Medical Center, Rotterdam (The Netherlands) and it is publicly available on-line at the GEO (*Gene Expression Omnibus*) web site[4].

The second data set (herein, referred to as *HC-Salamanca*) contains the gene expression profiles in samples of bone marrow from 62 adult patients with AML plus a group of 6 healthy individuals. The gene chip Affymetrix U133A was also used. The data comes from the Hematology Service of the University Hospital of Salamanca (Spain).

Patients were classified according to the WHO (*World Health Organization*) classification into 4 subgroups: a) acute promyelocytic leukemia (APL) with recurrent cytogenetic translocation of type t(15;17), b) AML with recurrent cytogenetic translocation of type inv(16), c) acute monocytic leukemia and d) other AML not well-characterized. The class assignment of the profiles to a particular subgroup of AML is not available for 11 patients in the *EUMC-Rotterdam* data set. The distribution of the two data sets according to this classification is shown in Table 6.1.

Only the microarrays from the three well-characterized AML subtypes and the control group (healthy persons) have been used in the experiments. Deal with the

[4] http://www.ncbi.nlm.nih.gov/projects/geo/

Table 6.1. Class assignment of gene expression profiles within available data sets

Data set	*EUMC-Rotterdam*	*HC-Salamanca*
Healthy persons	8	6
APL with t(15;17)	19	10
AML with inv(16)	14	4
Monocytic	64	7
Other AML	177	41
Not available	11	0

microarrays from the "Other AML" group requires an unsupervised technique in order to clustering similar microarrays and, if some biological evidence supports it, discover new subtypes of AML. Since our proposed technique is supervised, it seems reasonable to deal only with gene expression profiles from groups which are well classified. Therefore, the *EUMC-Rotterdam* and *HC-Salamanca* data sets are reduced to 105 and 27 samples, respectively.

Before the experimentation with those microarrays, the original data sets must be preprocessed in order to perform a normalization of the available data. Both data sets have been normalized using the RMA (*Robust Multichip Average*) algorithm [3], which is proposed by Affymetrix to perform a background adjustment, a quantile normalization, and finally a summarization.

6.5.2 Results

The employed methodology splits the available data within a test set and a training set. Concretely, the *HC-Salamanca* data set was used to test the GCS network trained with the available microarrays in the *EUMC-Rotterdam* data set.

The proposed technique consists of two differentiated steps. Firstly, the filtering of superfluous genes by mean of the computation of the DFP, and secondly, the clustering process by mean of the training of a GCS network. Both processes require the assessment of the θ and π parameters, and the maximum number of nodes in the GCS network. The two first parameters determine the length of the DFP and, therefore, the number of meaningful genes which are selected. The last parameter determines the capacity of clustering of the GCS network.

Over the training data set, a strategy of cross-validation is used in order to asses these parameters, concretely, a 3-fold stratified cross-validation. In each round, each fold of the original training set (which acts as an evaluation set) is used to estimate the predictive accuracy of the GCS network (referred to as evaluation error) trained from the rest of folds.

Different configurations of θ and π values have been evaluated. For each one it has been computed the mean of evaluation error for each evaluation data set. Specifically, we have been evaluated the configurations of the Cartesian product $\{0.75, 0.8, 0.85, 0.9, 0.95\}\times\{0.65, 0.7, 0.75, 0.8, 0.85\}$ for the parameters θ and π, respectively, and for a maximum number of nodes in the GCS network ranging from 5 to 8.

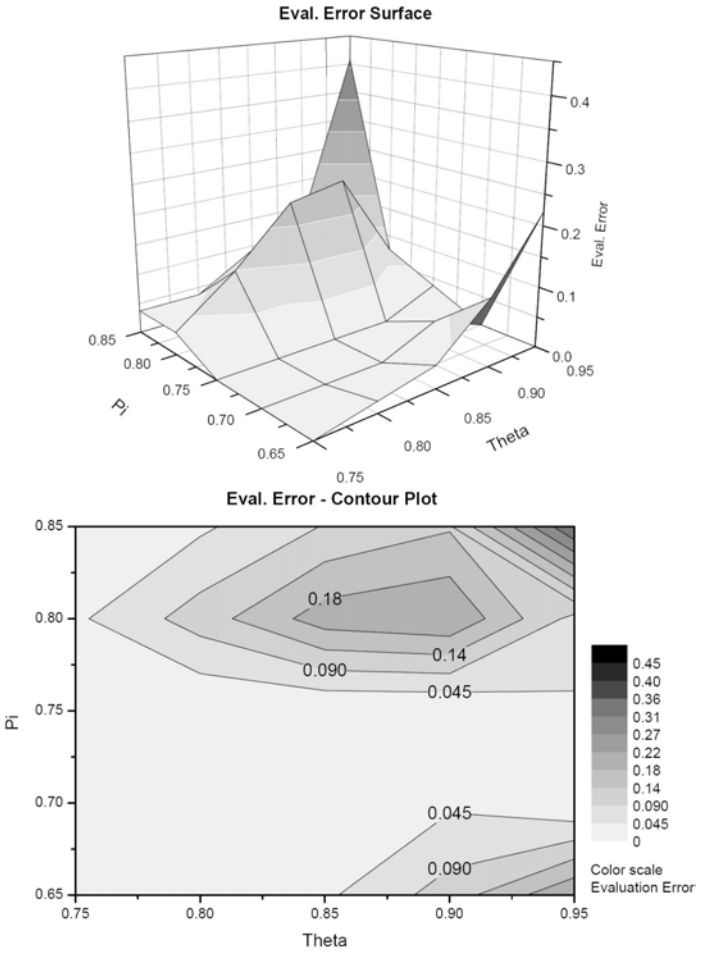


Fig. 6.9. Evaluation error vs. θ and π parameters

The evaluation error surface inferred from the experimental results, which depends on the values of θ and π parameters, is shown in Figure 6.9 (upper panel). The contour plot presented in Figure 6.9 (bottom panel) shows the existence of several configurations with minimal evaluation error, basically in the quadrant $\{0.75, 0.8, 0.85\}\times\{0.65, 0.7, 0.75\}$.

To refine the assessment of values to the relevant parameters of the proposed technique, the evaluation error is shown in Figure 6.10. These values depend on the maximum number of nodes considered in the training process of the GCS network. The upper graph shows the variation of the evaluation error for different levels of the θ parameter, whereas the lower graph shows the variation of the

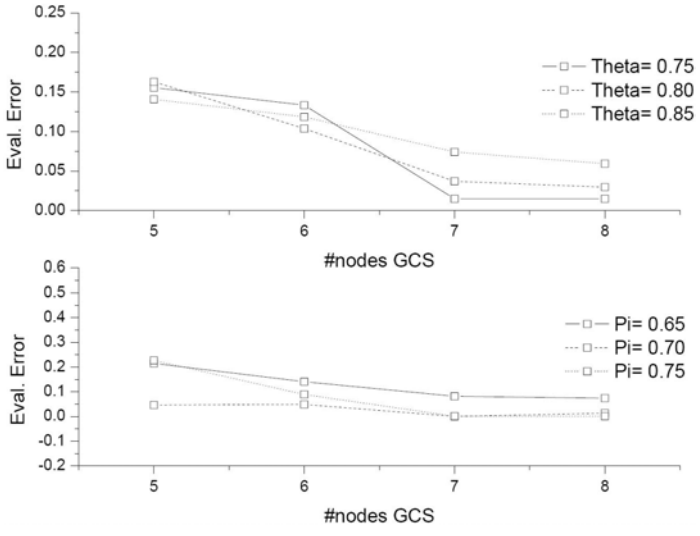


Fig. 6.10. Evaluation error vs. number of nodes of the GCS network

Table 6.2. Results of the three tested configurations

Θ	Π	#featuresin DFP	#nodes GCS	TrainingError	TestError
0.75	0.70	469	7	15/105	0/27
0.75	0.75	258	7	13/105	0/27
0.75	0.75	258	8	13/105	0/27

error for different π values. As it can be seen on Figure 6.10, the configurations $\{(0.75, 0.7, 7), (0.75, 0.75, 7), (0.75, 0.7, 8)\}$ seem to be a "good" choice of parameters θ, π and maximum number of nodes in GCS, respectively.

Starting from the previous experimentation, Table 6.2 shows the results of the tested configurations taking into account the number of selected genes, the training error and the test error. In next section we discuss the major issues about the obtained results in the experimentation process.

6.5.3 Discussion

Firstly, we introduce some comments about the specific results obtained in the experimentation which has been carried out, and finally, we conclude this section with general remarks about the proposed technique.

The parameter θ determines the granularity of the fuzzy discretization which is carried out by the computation of fuzzy patterns for each class. Therefore, it is quite related to the expressive power of the implicit knowledge representation imposed by linguistic labels over the numerical data of original microarrays.

Since the three tested configurations have the same value for parameter θ, now we are interested in the concrete value of π parameter.

The π parameter controls the frequency threshold that is required to include a gene in the corresponding FP of a class, and consequently, the length of the DFP representative for all the classes. Given a fixed level for the θ parameter, a higher value of π parameter implies a lower number of genes belonging to DFP, and therefore, a lower number of meaningful genes selected in the first step of the proposed technique. The number of genes selected for the considered configurations were 469 (with $\pi = 0.7$) and 258 (with $\pi = 0.8$). The complete list of genes is not shown due to space limitations.

As it can be seen in Table 6.2, the training error over the *EUMC-Rotterdam* data set is about the 12-14% whereas the test error over the *HC-Salamanca* data set is 0%.

Given the three tested configurations, we prefer the second one since it leads to a less complex model (in terms of the selected features and the number of nodes of the GCS network which is trained). Simultaneously, this configuration produces a model which maintains the higher accuracy reached for all the tested scenarios. Moreover, and since the origin of the two data sets are completely independent, the 0% error over test data can show the high capacity of generalization of the proposed technique. This characteristic is always a desirable property for any model.

Summarizing, this work explores the application of fuzzy logic to the process of gene selection and data reduction in the microarray data domain. In this sense, we have applied fuzzy logic to discretize the original data about the expression level of a gene within three linguistic labels (LOW, MEDIUM and HIGH). This fact leads to the possibility of clearly identifying the genes (those one belonging to the discriminant fuzzy pattern), which are meaningful to discriminate between patients and classes.

The work also explores the capabilities of a growing cell structure neural network to discover relevant knowledge for clustering patients suffering for acute myeloid leukemia. A key advantage of the proposed method is that it allows incorporating biological meaningful information to the network operation in the form of a gene-based distance metric. Moreover, the GCS network makes use of a previous successful fuzzy discretization method for data reduction on microarray data domain.

Using self-organizing GCS networks to meaningfully cluster filtered microarray data has a number of appealing features over other approaches (i.e. incremental self-construction and easy visualization of biological relationships among the input data). The explanations of the clustering process carried out by the network can be addressed by means of our DFP vector. The most relevant knowledge for each cluster can be highlighted, and provide meaningful explanations about the clustering process and useful insights into the underlying problem and data. The experimental results show that with only a small subset of the genes belonging to a sample, the performance of the network in terms of the clustering accuracy rate rises to 100%.

Acknowledgments

We are deeply indebted to the personnel at the Hematology Service of the University Hospital of Salamanca and Cancer Research Center, who made this research work possible, by providing data and allowing us access to facilities.

References

1. Alizadeh, A.A., Eisen, M.B., Davis, R.E., Ma, C., Lossos, I.S., Lossos, A., Rosenwald, J., Boldrick, J.C., Sabet, H., Tran, T., Yu, X., Powell, J.I., Yang, L., Marti, G.E., Moore, T., Hudson Jr., J., Lu, L., Lewis, D.B., Tibshirani, R., Sherlock, G., Chan, W.C., Greiner, T.C., Weisenburger, D.D., Armitage, J.O., Warnke, R., Levy, R., Wilson, W., Grever, M.R., Byrd, J.C., Botstein, D., Brown, P.O., Staudt, L.M.: Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
2. Blanco, R., Larrañaga, P., Inza, I., Sierra, B.: Gene selection for cancer classification using wrapper approaches. International Journal of Pattern Recognition and Artificial Intelligence 18(8), 1373–1390 (2004)
3. Bolstad, B.M., Irizarry, R.A., Astrand, M., Speed, T.P.: A comparison of normalization methods for high density oligonucleotide array data based on bias and variance. Bioinformatics (2), 185–193 (2003)
4. Chu, F., Wang, L.: Applications of support vector machines to cancer classification with microarray data. International Journal of Neural Systems 15(6), 475–484 (2005)
5. Chu, F., Wang, L.: Gene expression data analysis using support vector machines. In: Bioinformatics using Computational Intelligence Paradigms, pp. 167–189. Springer, Berlin (2005)
6. Dai, J.J., Lieu, L., Rocke, D.: Dimension reduction for classification with gene expression microarray data. Statistical Applications in Genetics and Molecular Biology 5(1), 6 (2006)
7. Deutsch, J.M.: Evolutionary algorithms for finding optimal gene sets in microarray prediction. Bioinformatics 19(1), 45–52 (2003)
8. Díaz, F., Fdez-Riverola, F., Corchado, J.M.: Gene-cbr: A case-based reasoning tool for cancer diagnosis using microarray datasets. Computational Intelligence 22(3-4), 254–268 (2006)
9. Díaz, F., Fdez-Riverola, F., Glez-Peña, D., Corchado, J.M.: Using fuzzy patterns for gene selection and data reduction on microarray data. In: Proceedings of the 7th International Conference on Intelligent Data Engineering and Automated Learning, Burgos, Spain, pp. 1087–1094 (2006)
10. Dubois, D., Prade, H.: Fuzzy sets and systems: Theory and applications. Academic Press, New York (1980)
11. Fdez-Riverola, F., Díaz, F., Borrajo, M.L., Yánez, J.C., Corchado, J.M.: Improving gene selection in microarray data analysis using fuzzy patterns inside a cbr system. In: Proceedings of the 6th International Conference on Case-Based Reasoning, Chicago, Illinois, USA, pp. 191–205 (2005)
12. Fritzke, B.: Growing self-organising networks – why? In: Proceedings of the 11th European Symposium on Artificial Neural Networks, pp. 61–72 (1993)
13. Fritzke, B.: Growing cell structures - a self-organizing network for unsupervised and supervised learning, Tech. report, International Computer Science Institute, Berkeley, CA, USA (1993b)

14. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Collar, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
15. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46(1-3), 389–422 (2002)
16. Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies 7(1), 1–13 (1975)
17. Hanczar, B., Courtine, M., Benis, A., Hennegar, C., Clément, K., Zucker, J.D.: Improving classification of microarray data using prototype-based feature selection. ACM SIGKDD Explorations Newsletter 5(2), 23–30 (2003)
18. Hochreiter, S., Obermayer, K.: Feature selection and classification on matrix data: from large margins to small covering numbers. In: Advances in Neural Information Processing Systems, vol. 15, pp. 913–920. MIT Press, Cambridge (2003)
19. Jaeger, J., Sengupta, R., Ruzzo, W.L.: Improved gene selection for classification of microarrays. In: Proceedings of the 8th Pacific Symposium on Biocomputing, Kauai, Hawaii, pp. 53–64 (2003)
20. Jang, J.S.R., Sun, C.T.: Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. Prentice-Hall, Englewood Cliffs (1997)
21. Khan, J., Wei, J.S., Ringnér, M., Saal, L.H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C.R., Peterson, C., Meltzer, P.S.: Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nature Medicine 7(6), 673–679 (2001)
22. Kohonen, T.: Self-organising maps. Springer, Berlin (1995)
23. Li, L., Darden, T.A., Weinberg, C.R., Levine, A.J., Pedersen, L.G.: Gene assessment and sample classification for gene expression data using a genetic algorithm/k-nearest neighbor method. Combinatorial Chemistry & High Throughput Screening 4(8), 727–739 (2001)
24. Liu, B., Wan, C., Wang, L.: Unsupervised gene selection via spectral biclustering. In: Proceedings of IEEE International Joint Conference on Neural Networks, Budapest, Hungary, pp. 1681–1686 (2004)
25. Niijima, S., Kuhara, S.: Effective nearest neighbor methods for multiclass cancer classification using microarray data. In: Proceedings of the 16th International Conference on Genome Informatics, p. P051 (2005)
26. Ochs, M.F., Godwin, A.K.: Microarrays in cancer: Research and applications. BioTechniques 34, s4–s15 (2003)
27. Valk, P.J., Verhaak, R.G., Beijen, M.A., Erpelinck, C.A., van Waalwijk, B., Doorn-Khosrovani, S., Boer, J.M., Beverloo, H.B., Moorhouse, M.J., van der Spek, P.J., Löwenberg, B., Delwel, R.: Prognostically useful gene-expression profiles in acute myeloid leukaemia. New England Journal of Medicine 350(16), 1617–1628 (2004)
28. Qi, H.: Feature selection and knn fusion in molecular classification of multiple tumor types. In: Proceedings of the International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences, Las Vegas, Nevada, USA (1992)
29. Sugeno, M.: Industrial applications of fuzzy control. Elsevier, Amsterdam (1985)
30. Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., Vapnik, V.: Feature selection for svms. In: Advances in Neural Information Processing Systems, vol. 13, pp. 668–674. MIT Press, Cambridge (2001)
31. Zadeh, L.A.: Fuzzy sets. Information and Control 12, 338–353 (1965)

32. Zadeh, L.A.: Soft computing and fuzzy logic. IEEE Software 11(6), 48–56 (1994)
33. Zheng, B., Olusegun, E., Narasimhan, G.: Neural network classifiers and gene selection methods for microarray data on human lung adenocarcinoma. In: Proceedings of the 6th Critical Assessment of Microarray Data Analysis, North Carolina, USA, pp. 63–67 (2003)
34. Zong, N., Adjouadi, M., Ayala, M.: Optimizing the classification of acute lymphoblastic leukemia and acute myeloid leukemia samples using artificial neural networks. Biomedical Sciences Instrumentation 42, 261–266 (2006)

7

Gene Expression Analysis by Fuzzy and Hybrid Fuzzy Classification

Gerald Schaefer[1], Tomoharu Nakashima[2], and Hisao Ishibuchi[2]

[1] School of Engineering and Applied Science, Aston University, Birmingham, U.K.
G.Schaefer@aston.ac.uk
[2] Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Osaka, Japan
{nakashi,hisaoi}@cs.osakafu-u.ac.jp

Summary. Microarray studies and gene expression analysis has received tremendous attention over the last few years and provide many promising avenues towards the understanding of fundamental questions in biology and medicine. In this chapter we show that the employment of a fuzzy rule-based classification system allows for effective analysis of gene expression data. The applied classifier consists of a set of fuzzy if-then rules that allows for accurate non-linear classification of input patterns. We further show that a hybrid fuzzy classification scheme in which a small number of fuzzy if-then rules are selected through means of a genetic algorithm is capable of providing a compact classifier for gene expression analysis. Extensive experimental results on various well-known gene expression databsets confirm the efficacy of the presented approaches.

7.1 Introduction

Microarray expression studies measure, through a hybridisation process, the levels of genes expressed in biological samples. Knowledge gained from these studies is deemed increasingly important due to its potential of contributing to the understanding of fundamental questions in biology and clinical medicine. Microarray experiments can either monitor each gene several times under varying conditions or analyse the genes in a single environment but in different types of tissue. In this chapter we focus on the latter where one important aspect is the classification of the recorded samples. This can be used to either categorise different types of cancerous tissues as in [7] where different types of leukemia are identified, or to distinguish cancerous tissue from normal tissue as done in [2] where tumor and normal colon tissues are analysed.

One of the main challenges in classifying gene expression data is that the number of genes is typically much higher than the number of analysed samples. Also is it not clear which genes are important and which can be omitted without reducing the classification performance. Many pattern classification techniques have been employed to analyse microarray data. For example, Golub *et al.* [7] used a weighted voting scheme, Fort and Lambert-Lacroix [5] employed partial least squares and logistic regression techniques, whereas Furey *et al.* [6] applied

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 127–140.
springerlink.com

support vector machines. Dudoit *et al.* [4] investigated nearest neighbour classifiers, discriminant analysis, classification trees and boosting, while Statnikov *et al.* [15] explored several support vector machine techniques, nearest neighbour classifiers, neural networks and probabilistic neural networks. In several of these studies it has been found that no one classification algorithm is performing best on all datasets (although for several datasets SVMs seem to perform best) and that hence the exploration of several classifiers is useful. Similarly, no universally ideal gene selection method has yet been found as several studies [14, 15] have shown.

In this chapter we apply fuzzy rule-based classification concepts to the classification of microarray expression data and show, based on a series of experiments, that it affords good classification performance for this type of problem. Several authors have used fuzzy logic to analyse gene expression data before. Woolf and Wang [18] used fuzzy rules to explore the relationships between several genes of a profile while Vinterbo *et al.* [17] used fuzzy rule bases to classify gene expression data. However, Vinterbo's method has the disadvantage that it allows only linear discrimination. Furthermore, they describe each gene by only 2 fuzzy partitions ('up' and 'down') while we also explore division into more intervals and show that by doing so increased classification performance is possible.

Based on this fuzzy rule classification system we then show that a compact rule base can be extracted using a genetic algorithm that assesses the fitness of individual rules and selects a rule ensemble that maximises classification performance while maintining computational efficiency due to a preset small size of the rule base.

7.2 Fuzzy Rule-Based Classification

While in the past fuzzy rule-based systems have been mainly applied to control problems [16], more recently they have also been applied to pattern classification problems. Various methods have been proposed for the automatic generation of fuzzy if-then rules from numerical data for pattern classification and have been shown to work well on a variety of problem domains [8, 11, 10].

Pattern classification typically is a supervised process where, based on set of training samples with known classifications, a classifier is derived that performs automatic assignment to classes based on unseen data. Let us assume that our pattern classification problem is an n-dimensional problem with C classes (in microarray analysis C is often 2) and m given training patterns $\mathbf{x}_p = (x_{p1}, x_{p2}, \ldots, x_{pn})$, $p = 1, 2, \ldots, m$. Without loss of generality, we assume each attribute of the given training patterns to be normalised into the unit interval $[0, 1]$; that is, the pattern space is an n-dimensional unit hypercube $[0, 1]^n$. In this study we use fuzzy if-then rules of the following type as a base of our fuzzy rule-based classification systems:

$$\begin{array}{l}\text{Rule } R_j\text{: If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn} \\ \text{then Class } C_j \text{ with } CF_j, \quad j = 1, 2, \ldots, N,\end{array} \tag{7.1}$$

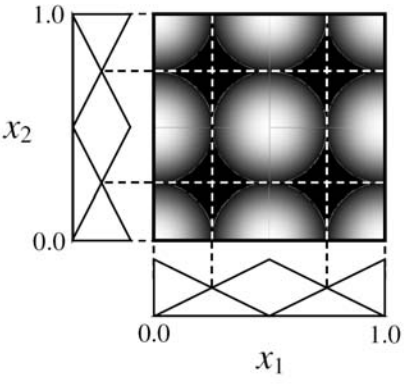


Fig. 7.1. Example triangular membership function ($L = 3$)

where R_j is the label of the j-th fuzzy if-then rule, $A_{j1}, \ldots, A_{jn}$ are antecedent fuzzy sets on the unit interval $[0, 1]$, C_j is the consequent class (i.e., one of the C given classes), and CF_j is the grade of certainty of the fuzzy if-then rule R_j. As antecedent fuzzy sets we use triangular fuzzy sets as in Figure 7.1 where we show the partitioning of two variables into a number of fuzzy sets.

Our fuzzy rule-based classification system consists of N linguistic rules each of which has a form as in Equation (7.1). There are two steps in the generation of fuzzy if-then rules: specification of antecedent part and determination of consequent class C_j and the grade of certainty CF_j. The antecedent part of fuzzy if-then rules is specified manually. Then, the consequent part (i.e., consequent class and the grade of certainty) is determined from the given training patterns [13]. In [12] it is shown that the use of the grade of certainty in fuzzy if-then rules allows us to generate comprehensible fuzzy rule-based classification systems with high classification performance.

7.2.1 Fuzzy Rule Generation

Let us assume that m training patterns $\mathbf{x}_p = (x_{p1}, \ldots, x_{pn})$, $p = 1, \ldots, m$, are given for an n-dimensional C-class pattern classification problem. The consequent class C_j and the grade of certainty CF_j of the if-then rule are determined in the following two steps:

1. Calculate $\beta_{\text{Class } h}(j)$ for Class h as

$$\beta_{\text{Class } h}(j) = \sum_{\mathbf{x}_p \in \text{Class } h} \mu_j(\mathbf{x}_p), \tag{7.2}$$

where

$$\mu_j(\mathbf{x}_p) = \mu_{j1}(x_{p1}) \cdot \ldots \cdot \mu_{jn}(x_{pn}), \tag{7.3}$$

and $\mu_{jn}(\cdot)$ is the membership function of the fuzzy set A_{jn}. In this chapter we use triangular fuzzy sets as in Figure 7.1.

2. Find Class $\hat{h}$ that has the maximum value of $\beta_{\text{Class } h}(j)$:

$$\beta_{\text{Class } \hat{h}}(j) = \max_{1 \le k \le C} \{\beta_{\text{Class } k}(j)\}. \tag{7.4}$$

If two or more classes take the maximum value, the consequent class C_j of the rule R_j can not be determined uniquely. In this case, specify C_j as $C_j = \phi$. If a single class $\hat{h}$ takes the maximum value, let C_j be Class $\hat{h}$. The grade of certainty CF_j is determined as

$$CF_j = \frac{\beta_{\text{Class } \hat{h}}(j) - \bar{\beta}}{\sum_h \beta_{\text{Class } h}(j)} \tag{7.5}$$

with

$$\bar{\beta} = \frac{\sum_{h \neq \hat{h}} \beta_{\text{Class } h}(j)}{C - 1}. \tag{7.6}$$

7.2.2 Fuzzy Reasoning

Using the rule generation procedure outlined above we can generate N fuzzy if-then rules as in Equation (7.1). After both the consequent class C_j and the grade of certainty CF_j are determined for all N rules, a new pattern $\mathbf{x} = (x_1, \ldots, x_n)$ can be classified by the following procedure:

1. Calculate $\alpha_{\text{Class } h}(\mathbf{x})$ for Class h, $j = 1, \ldots, C$, as

$$\alpha_{\text{Class } h}(\mathbf{x}) = \max\{\mu_j(\mathbf{x}) \cdot CF_j | C_j = h\}, \tag{7.7}$$

2. Find Class h' that has the maximum value of $\alpha_{\text{Class } h}(\mathbf{x})$:

$$\alpha_{\text{Class } h'}(\mathbf{x}) = \max_{1 \le k \le C} \{\alpha_{\text{Class } k}(\mathbf{x})\}. \tag{7.8}$$

If two or more classes take the maximum value, then the classification of $\mathbf{x}$ is rejected (i.e. $\mathbf{x}$ is left as an unclassifiable pattern), otherwise we assign $\mathbf{x}$ to Class h'.

7.2.3 Rule Splitting

It is generally known that any type of rule-based system suffers from the curse of dimensionality. That is, the number of generated rules increases exponentially with the number of attributes involved. Our fuzzy rule-based classifier is no exception, in particular considering that for successful classification of microarray data typically at least a few dozens genes are selected. For example, based on the selection of 50 genes, the classifier would generate $2^{50} = 1.1259 * 10^{15}$ rules even if we only partition each axis into two which is clearly prohibitive both in terms of storage requirements and computational complexity. We therefore apply a rule splitting step and limit the number of attributes in a fuzzy if-then rule to 2. As the number of combinations of attribute pairs is $\binom{50}{2} = 1225$ for 50 genes and as for two fuzzy sets for each attribute $2^2 = 4$ rules are necessary in total we need only $4 * 1225 = 4900$ rules, a number significantly lower than 2^{50}.

7.3 Hybrid Fuzzy Classification

Even though the classifier developed in Section 7.2 has been shown to work well on various pattern classification problems [12] its major drawback is that due to the exhaustive way of generating the underlying rule base, the number of generated rules is high, even for low dimensional feature vectors. As mentioned in Section 7.2.3 one way of reducing the complexity of the classifier is to use only rules with 2 attributes. However, this approach still generates a high number of rules for higher dimensional data. In the following we show how, through application of a genetic algorithm (GA) [9], a compact fuzzy rule base for classification can be derived.

The fuzzy if-then rules that we are using in this approach are of the same form as the one given in Equation (7.1), i.e. contain a number of fuzzy attributes and a consequent class together with a grade of certainty. Our approach of using genetic algorithms to generate a fuzzy rule-based classification system is a Michigan style algorithm [10] which represents each linguistic rule by a string and handles it as an individual in the population of the GA. A population consists of a pre-specified number of rules. Because the consequent class and the rule weight of each rule can be easily specified from the given training patterns as shown in Section 7.2, they are not used in the coding of each linguistic rule (i.e., they are not included in a string). Each rule is represented by a string using its antecedent fuzzy sets.

7.3.1 Genetic Operations

First the algorithm randomly generates a pre-specified number N_{rule} of rules as an initial population (in our experiments we set $N_{\text{rule}} = 20$). Next the fitness value of each linguistic rule in the current population is evaluated. Let S be the set of rules in the current population. The evaluation of each rule is performed by classifying all the given training patterns by the rule set S using the single winner-based method described in Section 7.2. The winning rule receives a unit reward when it correctly classifies a training pattern. After all the given training patterns are classified by the rule set S, the fitness value $fitness(R_q)$ of each linguistic rule R_q in S is calculated as

$$fitness(R_q) = NCP(R_q), \tag{7.9}$$

where $NCP(R_q)$ is the number of correctly classified training patterns by R_q. It should be noted that the following relation holds between the classification performance $NCP(R_q)$ of each linguistic rule R_q and the classification performance $NCP(S)$ of the rule set S used in the fitness function:

$$NCP(S) = \sum_{R_q \in S} NCP(R_q). \tag{7.10}$$

The algorithm is implemented so that only a single copy is selected as a winner rule when multiple copies of the same linguistic rule are included in the rule

set S. In GA optimisation problems, multiple copies of the same string usually have the same fitness value. This often leads to undesired early convergence of the current population to a single solution. In our algorithm, only a single copy can have a positive fitness value and the other copies have zero fitness which prevents the current population from being dominated by many copies of a single or few linguistic rules.

Then, new rules are generated from the rules in the current population using genetic operations. As parent strings, two fuzzy if-then rules are selected from the current population and binary tournament selection with replacement is applied. That is, two rules are randomly selected from the current population and the better rule with the higher fitness value is chosen as a parent string. A pair of parent strings is chosen by iterating this procedure twice.

From the selected pair of parent strings, two new strings are generated by a crossover operation. We use a uniform crossover operator, illustrated in Figure 7.2 where crossover positions (indicated by "$*$") are randomly chosen for each pair of parent strings. The crossover operator is applied to each pair of parent strings with a pre-specified crossover probability p_c. After new strings are generated, each symbol of the generated strings is randomly replaced with a different symbol by a mutation operator with a pre-specified mutation probability p_m. Usually the same mutation probability is assigned to every position of each string. The mutation operator is illustrated in Figure 7.3 where mutated values are underlined. Selection, crossover, and mutation are iterated until a pre-specified number N_{replace} of new strings are generated.

Finally, the N_{replace} strings with the smallest fitness values in the current population are removed, and the newly generated N_{replace} strings added to form a new population. Because the number of removed strings is the same as the number of added strings, every population consists of the same number of strings. That is, every rule set has the same number of linguistic rules. This generation update can be viewed as an elitist strategy where the number of elite strings is $(N_{\text{rule}} - N_{\text{replace}})$.

The above procedures are applied to the new population again. The generation update is iterated until a pre-specified stopping condition is satisfied. In our

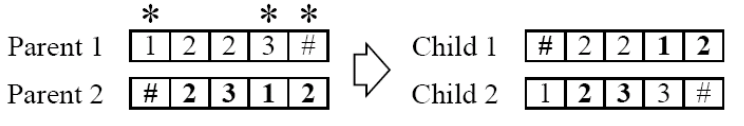


Fig. 7.2. Example of crossover operator

Fig. 7.3. Example of mutation operator

experiments we use the total number of iterations (i.e., the total number of generation updates) as stopping condition.

7.3.2 Algorithm Summary

To summarise, our hybrid fuzzy rule-based classifier works as follows:

Step 1: *Parameter Specification.* Specify the number of linguistic rules N_{rule}, the number of replaced rules N_{replace}, the crossover probability p_c, the mutation probability p_m, and the stopping condition.
Step 2: *Initialization.* Randomly generate N_{rule} rules (i.e., N_{rule} strings of length n) as an initial population.
Step 3: *Genetic Operations.* Calculate the fitness value of each rule in the current population. Generate N_{replace} rules using selection, crossover, and mutation from existing rules in the current population.
Step 4: *Generation Update (Elitist Strategy).* Remove the worst N_{replace} rules from the current population and add the newly generated N_{replace} rules to the current population.
Step 5: *Termination Test.* If the stopping condition is not satisfied, return to Step 3. Otherwise terminate the execution of the algorithm.

During the execution of the algorithm, we monitor the classification rate of the current population on the given training patterns. The rule set (i.e., population) with the highest classification rate is chosen as the final solution.

7.3.3 Increasing the Classification Performance

Randomly generated initial linguistic rules with fine fuzzy partitions usually do not classify many training patterns in high-dimensional pattern classification problems such as those encountered when analysing gene expression data. This is because each linguistic rule covers a very small portion of the pattern space.

A simple method for expanding the covered area by each initial rule is to increase the selection probability of "*don't care*" among the antecedent fuzzy sets. Let $p_{don't\ care}$ be the selection probability of *don't care* when initial linguistic rules are generated. In this case, the selection probability of each of the other five antecedent fuzzy sets is $(1 - p_{don't\ care})/5$. Thus the portion of the pattern space covered by each initial rule can be increased by increasing the selection probability of *don't care*. In turn, this simple trick has a significant positive effect on the search ability of the hybrid fuzzy classifier.

Another method for generating initial rules with high classification ability is to use training patterns for specifying their antecedent fuzzy sets [10]. To generate an initial population of N_{rule} fuzzy rules, first we randomly select N_{rule} training patterns. Next we choose the combination of the most compatible linguistic terms with each training pattern. Note that *don't care* is not used in this stage because any attribute values are fully compatible with *don't care* (i.e., because *don't care* is always chosen as the most compatible antecedent fuzzy set for any attribute values). Each linguistic term in the selected combination is replaced

with *don't care* using the selection probability $p_{don't\ care}$. The combination of the linguistic terms after this replacement is used as the antecedent part of an initial rule. This procedure is applied to all the randomly selected N_{rule} training patterns for generating an initial population of N_{rule} rules.

The specification of antecedent fuzzy sets from training patterns can be utilised not only for generating an initial population but also for updating the current population. When a training pattern is misclassified or its classification is rejected by the current population, the generation of a new fuzzy rule from the misclassified or rejected training pattern may improve the classification ability of the current population. We can modify the generation update procedure as follows. We generate a single linguistic rule using the genetic operations and another rule from a misclassified or rejected training pattern. When all the training patterns are correctly classified, two rules are generated using the genetic operations.

Another extension to the hybrid fuzzy algorithm is the introduction of a penalty term with respect to the number of misclassified training patterns to the fitness function in Equation (7.9) as follows:

$$fitness(R_q) = NCP(R_q) - w_{NMP} \cdot NMP(R_q), \tag{7.11}$$

where $NMP(R_q)$ is the number of misclassified training patterns and w_{NMP} is a positive constant. The fitness function in Equation (7.9) can be viewed as a special case of Equation (7.11) with $w_{NMP} = 0$. In Equation (7.11), $NCP(R_q)$ and $NMP(R_q)$ are calculated by classifying all the training patterns by the current population S including the linguistic rule R_q. To understand the effect of the second term of on the evolution of linguistic rules, let us consider a rule that correctly classifies ten patterns and misclassifies three patterns. If the misclassification penalty is zero (i.e., if $w_{NMP} = 0$), the fitness value of this rule is 10. Thus this rule is not likely to be removed from the current population. As a result, the three misclassified patterns will also be misclassified in the next population. On the other hand, the fitness value of this rule is negative (i.e., -5) when $w_{NMP} = 5$. In this case, the rule will be removed from the current population. As a result, the three misclassified patterns may be correctly classified by other rules or their classification may be rejected in the next population. From this we can see that the introduction of the misclassification penalty to the fitness function may improve the search ability of the algorithm to identify rule sets with high classification ability.

7.4 Classifying Gene Expression Data

To demonstrate the usefulness and efficacy of our algorithms we evaluated our proposed method on three gene expression data sets that are commonly used in the literature. In the following we characterise each dataset briefly:-

- Colon dataset [2]: This dataset is derived from colon biopsy samples. Expression levels for 40 tumor and 22 normal colon tissues were measured for

6500 genes using Affymetrix oligonucleotide arrays. The 2000 genes with the highest minimal intensity across the tissues were selected. We pre-process the data following [4], i.e. perform a thresholding [floor of 100 and ceil of 16000] followed by filtering [exclusion of genes with max/min < 5 and (max-min) < 500] and $\log_{10}$ transformation.

- Leukemia dataset [7]: Bone marrow or peripheral blood samples were taken from 47 patients with acute lymphoblastic leukemia (ALL) and 25 patients with acute myeloid leukemia (AML). The ALL cases can be further divided into 38 B-cell ALL and 9 T-cell ALL samples and it is this 3-class division that we are basing our experiments on rather than the simpler 2-class version which is more commonly referred to in the literature. Each sample is characterised by 7129 genes whose expression levels where measured using Affymetrix oligonucleotide arrays. The same preprocessing steps as for the Colon dataset are applied.
- Lymphoma dataset [1]: This dataset contains gene expression data of diffuse large B-cell lymphoma (DLBCL) which is the most common subtype of non-Hodgink's lymphome. In total there are 47 samples of which 24 are of germinal centre B-like" and the remaining 23 of activated B-like subtype. Each sample is described by 4026 genes, however there are many missing values. For simplicity we removed genes with missing values from all samples.

Although the datasets represent only 2-class or 3-class problems, due to the large number of genes involved any rule based classification system would consist of a very large number of rules and hence represent a fairly complex process. Also, not all genes are equally important for the classification task at hand. We therefore sort the significance of genes according to the BSS/WSS (the ratio of between group to within group sum of squares) criterion used in [4] and consider only the top 50 respectively 100 genes as input for our classification problem.

We perform standard leave-one-out cross-validation where classifier training is performed on all available data except for the sample to be classified and this process is performed for all samples[1]. Fuzzy rule based classifiers and hybrid fuzzy classifiers based on partition sizes L between 2 and 5 partitions for each gene were constructed. To evaluate the achieved results we also implemented nearest neighbour and CART classifiers. The nearest neighbour classifier we employ searches through the complete training data to identify the sample which is closest to a given test input and assigns the identified sample's class. CART [3] is a classical rule based classifier which builds a recursive binary decision tree based on misclassification error of subtrees.

The results on the three datasets are given in Tables 7.1 to 7.3. In each table we given the number of correctly classified samples (CR), the number of incorrectly classified or unclassified samples (FR), and the classification accuracy (Acc.), i.e. the percentage of correctly classified samples.

Looking at the results for the Colon dataset which are given in Table 7.1, for the case of 50 selected features the fuzzy classifier with 3 partitions performs

[1] It should be noted that the top 50 respectively 100 genes were selected solely based on the training set.

Table 7.1. Classification performance on Colon dataset given in terms of number of correctly classified samples (CR), falsely classified or unclassified samples (FR), and classification accuracy (Acc.). Results are given for leave-one-out cross validation. Experiments were performed with 50 and 100 selected genes respectively and with a varying number L of partitions per gene. For comparison results obtained using a nearest neighbour classifier and a rule-based CART classifier are also listed.

		test data		
n	classifier	CR	FR	Acc.
50	fuzzy $L = 2$	50	12	80.65
	fuzzy $L = 3$	**53**	**9**	**85.48**
	fuzzy $L = 4$	52	10	83.87
	fuzzy $L = 5$	48	14	77.42
	hybrid fuzzy $L = 2$	49.7	12.3	80.11
	hybrid fuzzy $L = 3$	52.0	10.0	83.87
	hybrid fuzzy $L = 4$	49.0	13.0	79.03
	hybrid fuzzy $L = 5$	50.0	12.0	80.64
	nearest neighbour	49	13	79.03
	CART	48	14	77.42
100	fuzzy $L = 2$	44	18	70.97
	fuzzy $L = 3$	51	11	82.26
	fuzzy $L = 4$	50	12	80.65
	fuzzy $L = 5$	46	16	74.19
	hybrid fuzzy $L = 2$	48.4	13.6	78.01
	hybrid fuzzy $L = 3$	49.7	12.3	80.11
	hybrid fuzzy $L = 4$	47.7	14.3	76.88
	hybrid fuzzy $L = 5$	48.0	14.0	77.42
	nearest neighbour	**52**	**10**	**83.87**
	CART	45	17	72.58

best with a classification accuracy of 85.48% which corresponds to 9 incorrectly classified cases while nearest neighbour classification and CART produce 13 and 14 errors respectively. However when selecting the 100 top genes the nearest neighbour classifier performs slightly better than the fuzzy system. It is interesting to compare the performance of the fuzzy rule-based classifier when using different numbers of partitions for each attribute. It can be seen that on this dataset the best performance is achieved when using 3 partitions (although on training data alone more partitions afford better performance). In particular it can be observed that the case with $L = 2$ as used in the work of Vinterbo *et al.* [17] produces the worst results and hence confirms that increasing the number of fuzzy intervals as we suggest leads to improved classification performance. However, it can also be seen that applying too many partitions can decrease classification performance as is apparent in the case of $L = 5$ on test data. For the hybrid fuzzy classifier we ran the experiment 10 times with different, random, initial populations and list the average of the 3 best runs. We can see the the hybrid fuzzy approach performs only slightly worse than the full fuzzy rule-based

Table 7.2. Classification performance on Leukemia dataset, laid out in the same fashion as Table 7.1

n	classifier	test data		
		CR	FR	Acc.
50	fuzzy $L = 2$	66	6	91.67
	fuzzy $L = 3$	68	4	94.44
	fuzzy $L = 4$	67	5	93.06
	fuzzy $L = 5$	66	6	91.67
	hybrid fuzzy $L = 2$	69.3	2.7	96.29
	hybrid fuzzy $L = 3$	69.3	2.7	96.29
	hybrid fuzzy $L = 4$	67.7	4.3	93.98
	hybrid fuzzy $L = 5$	65.7	6.3	91.20
	nearest neighbour	**70**	**2**	**97.22**
	CART	47	25	65.28
100	fuzzy $L = 2$	63	9	87.50
	fuzzy $L = 3$	**71**	**1**	**98.61**
	fuzzy $L = 4$	69	3	95.83
	fuzzy $L = 5$	67	5	93.06
	hybrid fuzzy $L = 2$	68.0	4.0	94.44
	hybrid fuzzy $L = 3$	67.3	4.7	93.52
	hybrid fuzzy $L = 4$	63.9	8.1	88.74
	hybrid fuzzy $L = 5$	63.0	9.0	87.50
	nearest neighbour	70	2	97.22
	CART	45	27	62.50

system which, considering that classification is performed based only on 20 selected rules, proves the potential of this method.

Turning our attention to the results on the Leukemia dataset which are given in Table 7.2 we see a similar picture. Again the worst performing fuzzy classifier is that which uses only two partitions per gene while the best performing one as assessed by leave-one-out cross validation is the case of $L = 3$. CART performs fairly poorly on this dataset with classification accuracies reaching only about 65% while nearest neighbour classification performs well again confirming previous observations that despite its simplicity nearest neighbour classifiers are well suited for gene expression classification [4]. The best classification results are achieved by the fuzzy classifier with $L = 3$ for the case of 100 selected genes with a classification accuracy of 98.61% and the nearest neighbour classifier with 97.22% for 50 selected genes. For the case of 50 features, the hybrid fuzzy classifier outperforms the conventional fuzzy classification system in most cases while for classification based on 100 features it is more accurate only for the case of $L = 2$.

Table 7.3 lists the results obtained from the Lymphoma dataset. Here, perfect classification s achieved by the fuzzy classifier with $L = 4$ for 50 selected genes and by nearest neighbour classification based on 100 genes. The hybrid fuzzy approach performs slightly worse in most cases but is significantly worse for

Table 7.3. Classification performance on Lymphoma dataset, laid out in the same fashion as Table 7.1

n	classifier	test data		
		CR	FR	Acc.
50	fuzzy $L = 2$	45	2	95.74
	fuzzy $L = 3$	46	1	97.87
	fuzzy $L = 4$	**47**	**0**	**100**
	fuzzy $L = 5$	44	3	93.62
	hybrid fuzzy $L = 2$	46.0	1.0	97.88
	hybrid fuzzy $L = 3$	43.3	3.7	92.20
	hybrid fuzzy $L = 4$	42.3	4.7	90.07
	hybrid fuzzy $L = 5$	43.0	4.0	91.49
	nearest neighbour	45	2	95.74
	CART	36	11	76.60
100	fuzzy $L = 2$	44	3	93.62
	fuzzy $L = 3$	44	3	93.62
	fuzzy $L = 4$	44	3	93.62
	fuzzy $L = 5$	39	8	82.98
	hybrid fuzzy $L = 2$	44.0	3.0	93.61
	hybrid fuzzy $L = 3$	42.1	4.9	89.65
	hybrid fuzzy $L = 4$	37.7	9.3	80.14
	hybrid fuzzy $L = 5$	34.7	12.3	73.76
	nearest neighbour	**47**	**0**	**100**
	CART	38	9	80.85

$L = 4$ and $L = 5$ and 100 features. The reason for this performance is probably that during the run, the genetic algorithm was unable to explore sufficiently all of the search space.

7.5 Conclusions

In this chapter we have demonstrated that fuzzy rule-based classification systems can be used effectively for the analysis of gene expression data. The presented classifier consists of a set of fuzzy if-then rules that allows for accurate non-linear classification of input patterns. Based on extensive experiments we confirmed that our fuzzy approach to classification affords good classification performance on a series of data sets. Furthermore, a hybrid fuzzy approach to classifying gene expression data was also presented which provides a much more compact rule based, achieved through the application of a genetic algorithm to select useful rules, compared to the conventional classifier. The classification performance of the hybrid method was shown to be comparable to that of the full classifier.

Acknowledgements

The first author would like to acknowledge the Daiwa Foundation and the Sasakawa Foundation for supporting this work.

References

1. Alizadeh, A.A., Eisen, M.B., Davis, E.E., Ma, C., Lossos, I.S., Rosenwald, A., Boldrick, J.C., Sabet, H., Tran, T., Yu, X., Powell, J.I., Yang, L., Marti, G.E., Moore, T., Hudson, J., Lu, L., Lewis, D.B., Tibshirani, R., Sherlock, G., Chan, W.C., Greiner, T.C., Weisenburger, D.D., Armitage, J.O., Warnke, R., Levy, R., Wilson, W., Grever, M.R., Byrd, J.C., Botstein, D., Brown, P.O., Staudt, L.M.: Different types of diffuse large B-cell lymphoma identified by gene expression profiles. Nature 403, 503–511 (2000)
2. 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. Proc. Natnl. Acad. Sci. USA 96, 6745–6750 (1999)
3. Breiman, L., Friedman, J.H., Olshen, R., Stone, R.: Classification and regression trees. Wadsworth (1984)
4. Dudoit, S., Fridlyand, J., Speed, T.P.: Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association 97(457), 77–87 (2002)
5. Fort, G., Lambert-Lacroix, S.: Classification using partial least squares with penalized logistic regression. Bioinformatics 21(7), 1104–1111 (2005)
6. Furey, T.S., Cristianini, N., Duffy, N., Bednarski, D.W., Schummer, M., Haussler, D.: Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics 16(10), 906–914 (2000)
7. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
8. Grabisch, M., Dispot, F.: A comparison of some methods of fuzzy classification on real data. In: 2nd Int. Conference on Fuzzy Logic and Neural Networks, pp. 659–662 (1992)
9. Holland, J.H.: Adaptation in natural and artificial systems. University of Mitchigan Press (1975)
10. Ishibuchi, H., Nakashima, T.: Improving the performance of fuzzy classifier systems for pattern classification problems with continuous attributes. IEEE Trans. on Industrial Electronics 46(6), 1057–1068 (1999)
11. Ishibuchi, H., Nakashima, T.: Performance evaluation of fuzzy classifier systems for multi-dimensional pattern classification problems. IEEE Trans. Systems, Man and Cybernetics - Part B: Cybernetics 29, 601–618 (1999)
12. Ishibuchi, H., Nakashima, T.: Effect of rule weights in fuzzy rule-based classification systems. IEEE Trans. Fuzzy Systems 9(4), 506–515 (2001)
13. Ishibuchi, H., Nozaki, K., Tanaka, H.: Distributed representation of fuzzy rules and its application to pattern classification. Fuzzy Sets and Systems 52(1), 21–32 (1992)

14. Liu, H., Li, J., Wong, L.: A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. Gene Informatics 13, 51–60 (2002)
15. Statnikov, A., Aliferis, C., Tsamardinos, I., Hardin, D., Levy, S.: A comprehensive evaluation of multicategory classification methods for microarray expression cancer diagnosis. Bioinformatics 21(5), 631–643 (2005)
16. Sugeno, M.: An introductory survey of fuzzy control. Information Science 30(1/2), 59–83 (1985)
17. Vinterbo, S.A., Kim, E.-Y., Ohno-Machado, L.: Small, fuzzy and interpretable gene expression based classifiers. Bioinformatics 21(9), 1964–1970 (2005)
18. Woolf, P.J., Wang, Y.: A fuzzy logic approach to analyzing gene expression data. Physiological Genomics 3, 9–15 (2000)

8

Detecting Gene Regulatory Networks from Microarray Data Using Fuzzy Logic

Guy N. Brock[1], Vasyl Pihur[1], and Laura Kubatko[2]

[1] University of Louisville, Louisville, KY, USA
guy.brock@louisville.edu
[2] The Ohio State University, Columbus, OH, USA
lkubatko@stat.osu.edu

Summary. With the arrival of high-throughput genomic data, biologists now have the ability to investigate the expression of genetic transcripts on a genome-wide scale. With this advancement, it is important to consider the regulation of gene expression in the context of a system, including the discovery of any genetic interactions that contribute to regulation. Genetic networks provide a concise representation of the interaction between multiple genes at the system level, giving investigators a broader view of the cellular state compared to a singular declaration of whether a gene is over/under expressed. Many methods currently exist to infer gene regulatory networks, including discrete models (Boolean networks, Bayesian networks), continuous models (weight matrices, differential equations models), and fuzzy logic models. The attractive feature of the fuzzy logic model is that it allows for a simplified rule structure, since observations are categorized, but retains information in the original data by allowing partial membership in multiple categories. The fuzzy logic model is flexible, and can be adapted to a variety of regulatory models and inferential rule sets. In this work, we review several recent advances in fuzzy logic methodologies developed for the genetic network reconstruction problem. The goals of the approaches range from whole genome screening of microarray data for small regulatory units, to detailed reconstruction of the iteractions between genes in a particular pathway. We apply the methods to real microarray data concerning the yeast cell cycle and simulated data concerning the Raf signaling pathway, and compare results with other well-known algorithms.

8.1 Introduction

The production of a viable protein from a gene is called gene expression, and the regulation of gene expression is a fundamental process necessary to maintain the viability of an organism. Therefore, knowledge of gene expression patterns and the genes involved in the regulation process is valuable information to scientific investigators. Understanding the modulation of a gene's expression pattern during normal cell and tissue functioning provides useful insights into the biological function of the gene. Investigators can infer how genes are affected by diseases by comparing expression patterns between diseased and non-diseased states. These studies also give indications about potential biological pathways to target for therapeutic drugs. Pharmaceutical companies can test how cells will react to new drug treatments by observing gene expression patterns

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 141–163.
springerlink.com

pre- and post- treatment. In short, studying gene expression patterns improves our understanding of biological systems, and also enhances our ability to combat disease and improve our quality of life.

One of the dilemmas encountered in investigating gene function and genetic interactions is the sheer number of genes that are contained within a genome. Yeast, a relatively simple organism when compared to humans, contains over six thousand genes in its genome. The estimated number of genes within the human genome is about thirty thousand [5, 56]. Clearly, experimentation on each gene individually would take a very long time. To overcome this, scientists have developed technologies which are scalable to the level of the genome, collectively known as DNA microarrays, to evaluate the behavior and inter-relationship between genes on a genomic scale. DNA microarrays enable investigators to take a 'snapshot' of the processes of the cell in a particular state by measuring the expression levels of thousands of genes simultaneously. This is done by quantifying the amount of messenger RNA (mRNA) for that gene which is contained within the cell.

Since the advancement of DNA microarray technology, there has been an explosion of methods developed for analyzing data of this type. The methodological areas range from experimental design [25], normalization [4], and missing value imputation [8], to cluster analysis [19], classification [40], identification of differentially expressed genes [18], and network modelling [51]. In addition to extensions and enhancements of classic statistical techniques, new methods involving fuzzy logic have been proposed to address many of these areas [2, 12, 32]. One area that is particularly challenging is the reverse engineering of gene regulatory networks using microarray data. Reconstruction of these regulatory networks gives biologists keen insight into how genes interact with each other, and the roles genes play in various biological pathways.

Several methods have been developed to construct genetic networks from gene expression data. These include Boolean networks [31, 1, 11], Bayesian networks [21, 27, 59, 12, 55], weight matrices [53], differential equations models [13, 17], causal inference [58], graphical Gaussian models [44], partial least squares [39], and fuzzy logic models [57, 42, 47, 36, 16, 7]. Though each method differs in its approach to the network reconstruction problem, the goal in every case is to determine how genes interact and regulate gene expression by relating gene expression levels of target genes with putative regulatory genes. Models which use the continuous data directly, such as weight matrices and differential equation models, often involve the estimation of numerous network parameters, which is difficult with the relative scarcity of samples typically available in microarray data. Boolean networks and Bayesian networks can simplify the data structure by converting continuous gene expression data into discrete data. However, this discretization of the data into one of two states, "on" or "off", may overlook valuable information. Fuzzy logic represents a compromise between discrete and continuous models, that can characterize the data in an intuitive manner but allows observations to maintain partial membership in multiple categories. The idea is to examine the data in a fashion that enables high-level,

human-like reasoning in the characterization of the regulatory network, while retaining much of the information in the original, continuous data.

In this work, we review several recent advances in fuzzy logic methodologies developed for the genetic network reconstruction problem. The goals of the approaches range from whole genome screening of microarray data for small regulatory units [57, 42, 7], to detailed reconstruction of the iteractions between genes in a particular pathway [47, 36, 16]. One of the primary advantages of fuzzy logic is the ability to translate numeric data into linguistic constructs, that can then be easily converted into testable hypotheses. In addition, the interpretability of fuzzy logic models can be aided by computationally powerful methods like neural networks [29, 37] and genetic algorithms [16]. The methods discussed in this chapter combine these attributes of interpretability and computational sophistication, providing attractive approaches for investigators to decipher gene regulatory networks.

The rest of this chapter is organized as follows. In Section 8.2, we give a brief overview of the mechanics behind the regulation of gene expression. We also illustrate how gene regulatory networks can be represented, and briefly discuss the technologies currently used to identify gene-gene and protein-protein interactions. In Section 8.3, we describe the general characteristics of algorithms for reverse engineering gene regulatory networks using fuzzy logic, and briefly review other commonly used methods as well. In Section 8.4, we demonstrate the effectiveness of the fuzzy logic network reconstruction algorithms on simulated data, and compare results with other well-known algorithms. In Section 8.5, we present results from application of the methods to real microarray data concerning the yeast cell cycle. Lastly, in Section 8.6 we give some concluding remarks and discuss future directions and extensions of the fuzzy logic algorithms for genetic network reconstruction.

8.2 Gene Regulatory Networks

The production of a protein from a gene, called gene expression, is a two step process. The first step is called transcription, where DNA (deoxyribonucleic acid) serves as a template to produce messenger RNA (ribonucleic acid), or mRNA. The second step is called translation. Here mRNA is transported outside the cell's nucleus into the cytoplasm, where it is translated into a polypeptide, forming a protein. Proteins are involved in the transport of molecules, structural integrity, communication between cells, and facilitation of chemical reactions.

The process of gene expression is regulated by certain proteins known as *transcription factors*. Transcription factors facilitate the transcription process by binding the DNA in the promoter or enhancer region of the gene, which is upstream of the gene's transcription start site. Binding of the transcription factor to the promoter region of the gene either enhances or represses the ability of RNA polymerase, the enzyme responsible for transcription, to bind and start the transcription process. Frequently, several transcription factors work collectively as a single complex, and attract additional intermediary proteins known as

cofactors that assist in recruitment of RNA polymerase and additional proteins necessary for transcription.

For a transcription factor to bind to the promoter region of a gene requires the conversion of the dense DNA structure, known as heterochromatin, to a more lightly packed form known as euchromatin. This is typically accomplished via chromatin remodeling factors or histone acetylation, and is occasionally linked to the transcription factors themselves [28]. The process exposes the upstream promoter region of the gene or genes of interest, and allows the transcription factors and affiliated complexes to bind there. Often, multiple genes are transcribed together, so that their expression is monitored in a coordinate fashion and regulated by a single protein known as the *operator*. The group of genes the operator regulates or stimulates is known as an *operon*. Thus, the regulation of any one gene's expression may be tied to the expression of multiple other genes, which often results in a complex process involving many interwoven pieces.

A genetic regulatory network can be represented graphically via a set of interconnected nodes, where each node on the graph represents a gene (or, in some cases, an operon), and lines connecting nodes on the graph represent regulatory interactions between genes. Figure 8.1 depicts the interactions between protein products of genes in the yeast cell cycle, as represented in the KEGG database [24]. Note that interactions between the nodes in the graph consist of several different types, including both positive and negative regulation, and protein-protein interaction versus direct transcriptional regulation. Regulatory networks can be decomposed into elementary subunits consisting of direct and indirect interactions (Figure 8.2, top, see [54]), and *network motifs*, which are building blocks of network architecture that occur in numbers that are

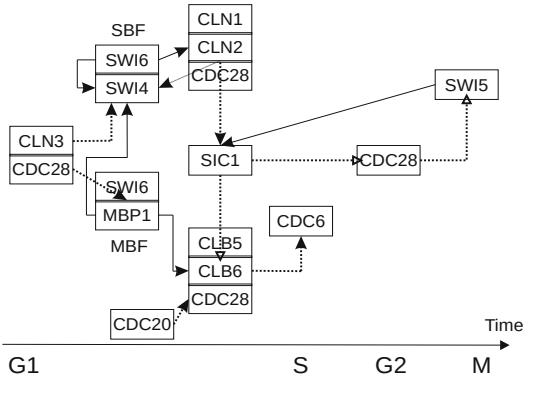


Fig. 8.1. Yeast cell cycle network, as given in the KEGG database. A filled arrow indicates a positive interaction, open arrows indicate negative interactions. Solid lines represent direct transcriptional regulation, dashed lines indicate protein-protein interactions. The time axis at the bottom of the figure refers to the portion of the yeast cell cycle for which the particular gene is active. The letters G1, S, G2, and M refer to distint periods in the yeast cell cycle.

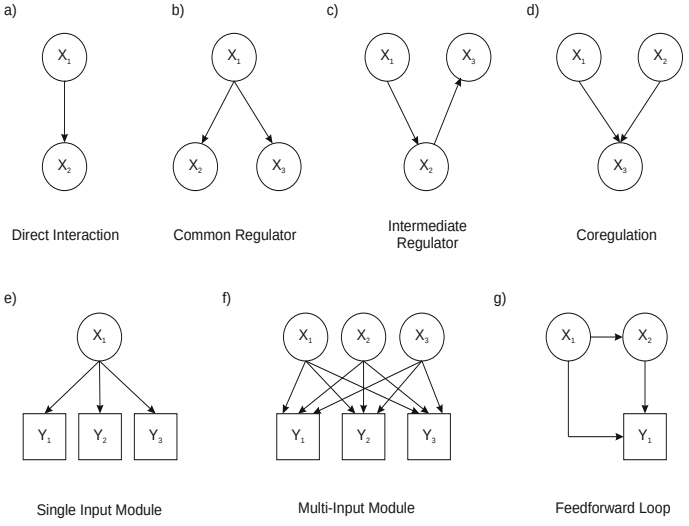


Fig. 8.2. Top: Direct (a) and indirect (b-d) interactions between network nodes, as depicted in [54]. Bottom: Commonly occuring regulatory motifs in the transcriptional regulation of *E. coli*, as reported in [46].

significantly higher than those in randomized blocks [30, 38, 46]. Figure 8.2, bottom, illustrates three motifs, feedforward loops, single input modules, and dense overlapping regions, that were found to occur frequently in the transcriptional networks of *E. coli* [46].

To study genetic networks, a variety of technologically advanced experimental techniques can be used to enable investigators to perform genome-wide studies of gene-gene and protein-protein interactions. The DNA-binding sites of transcription factors can be mapped on a genomic scale using ChiP-chip technology [41], which combines the concepts of chromatin immunoprecipation and DNA microarrays. Protein-protein interactions can be evaluated using affinity tags, the two-hybrid system, or quantitative proteomic techniques [3]. However, even these advanced experimental techniques for detecting physical interactions can produce a high number of false positives [22], and only a small number of interactions are supported by more than one method [52]. Hence, computational approaches for reverse engineering genetic association networks are extremely useful to biologists, as they provide a method for unraveling genetic interactions using existing high-throughput post-genomic data, and give important clues about where to focus future research efforts.

8.3 Fuzzy Logic Models for Genetic Networks

In this section, we review the general steps of the algorithms described in [7, 16, 47, 36, 57] for inferring gene regulatory networks from microarray gene expression

data. The general steps of most fuzzy algorithms are *fuzzification*, *inference*, and *defuzzification*. During fuzzification, gene expression values from the microarray data are assigned 'fuzzy' membership values in a set of fuzzy classes. Once the data are 'fuzzified', fuzzy rules relating the fuzzified gene expression levels to a specified target gene expression level are evaluated and selected. The selected fuzzy rules form a fuzzy rule set which is used to infer a fuzzy predicted target expression profile via fuzzy inference. Finally, the fuzzy predicted target expression profile is 'de-fuzzified' into a set of numbers on the same scale as the original microarray data. Networks can be ranked on the basis of a score which describes how well the predicted expression profile agrees with the original target expression profile, with regard to both the "closeness" to the observed value of the target for the data under consideration and in terms of the variety of configurations in which the collection of genes being considered was observed to occur.

8.3.1 A Fuzzy Logic Method to Screen for Network Triplets

Woolf and Wang [57] present a fuzzy logic model for network triplets consisting of an activator, repressor, and target gene. The model is conceptually simple, and allows screening of genetic interactions which would be missed by traditional clustering methods. Brock *et. al* [7] extend this approach by varying the number of states used in the model, as well as present an alternative model closely resembling the fuzzy logic model which is based on probability theory. Ressom *et. al* [42] also extend the approach of [57], by exploring alternate approaches for rule aggregation and defuzzification, and additionally speed the processing of the algorithm by implementing a clustering pre-conditioning step.

In each of these algorithms, gene expression values are first normalized to the $[0, 1]$ scale by subtracting the minimum of each gene's expression values and dividing by the range. In [57], the normalized expression values are then fuzzified using a three-state triangular matrix, with linguistic values "High", "Medium", and "Low". Brock *et. al* [7] explored the utility of increasing the number of states used to model the system, and found that a small to intermediate number of states had the best performance. Figure 8.3 depicts the membership functions for the three-state model, along with the membership values corresponding to a hypothetical activator / repressor gene pair.

After fuzzifying the data, a predicted target expression profile is determined for each activator / repressor gene combination. This is accomplished by using a decision matrix, an example of which is given in Table 8.1 for the three-state model. This decision matrix gives the fuzzy rules for determining target gene expression levels based on the input expression levels of the activator and repressor genes, in the form of IF-THEN clauses. For example, if the expression level for the activator gene (A) is High and the expression level for the repressor gene (R) is Low then the corresponding expression level for the hypothesized target gene (T) is High. This can be interpreted as a fuzzy statement or claim, and the degree of 'truth' associated with the claim 'T is High' is equal to the minimum of the degree of truth for the two antecedents in the claim, i.e. that 'A is High' and 'R is Low'. Here, the degree of truth for the claim 'A is High'

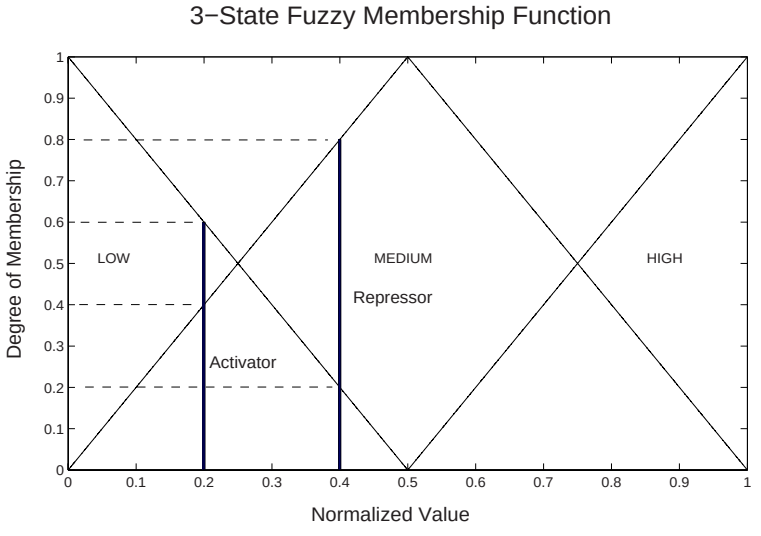


Fig. 8.3. Three-state fuzzy membership functions. Also depicted are example normalized expression levels for an activator and repressor gene, and their fuzzy representation.

Table 8.1. Decision matrix for the three-state fuzzy logic model. Entries within the decision matrix are the inferred levels of the target gene.

	Activator		
Repressor	Low	Medium	High
Low	Medium	High	High
Medium	Low	Medium	High
High	Low	Low	Medium

is synonymous with the fuzzy membership level that A has in the High state. If there are several claims where A and R predict T to be High, then the overall truth-value associated with the statement 'T is High' equals the maximum of all the truth-values for these individual claims.

Table 8.2. Illustration of the process of fuzzy inference

Antecedent		**Consequent**
IF x_A is Low (truth = 0.6)	AND x_R is Low (truth = 0.2)	THEN $\tilde{x}_T$ is Medium truth = min(0.6, 0.2) = 0.2
OR		
IF x_A is Medium (truth = 0.4)	AND x_R is Medium (truth = 0.8)	THEN $\tilde{x}_T$ is Medium truth = min(0.4, 0.8) = 0.4
Overall validity of claim that $\tilde{x}_T$ is Medium = max(0.2, 0.4) = 0.4		

Using the decision matrix and the fuzzified activator and repressor expression levels, the degree of membership that the target value has in each state is determined. This fuzzy target value can then be transformed back into a predicted value between 0 and 1 via the process of de-fuzzification. The method used for our fuzzy system is a zero-order Sugeno model [49], which uses a singleton output function that assigns a single value to each of the N fuzzy states in the model. The value for a particular fuzzy state is equal to its center of mass, obtained by interpreting the fuzzy membership function as a density function. To transform the fuzzified values back into numbers between 0 and 1, we take a weighted average of this output function, with the weights corresponding to the membership level that the fuzzy output has in each fuzzy state. Figure 8.4, originally given in+[7], illustrates the process, for an activator with membership levels 0.6 Low and 0.4 Medium, and a repressor with membership levels 0.2 Low and 0.8 Medium (neither gene has any membership in the High category). The resulting predicted target value is 0.389.

The fuzzy logic algorithm results in a predicted target expression profile for each of the $G(G-1)$ activator / repressor gene pairs, where G is the total number of genes. For a particular activator / repressor pair all of the remaining

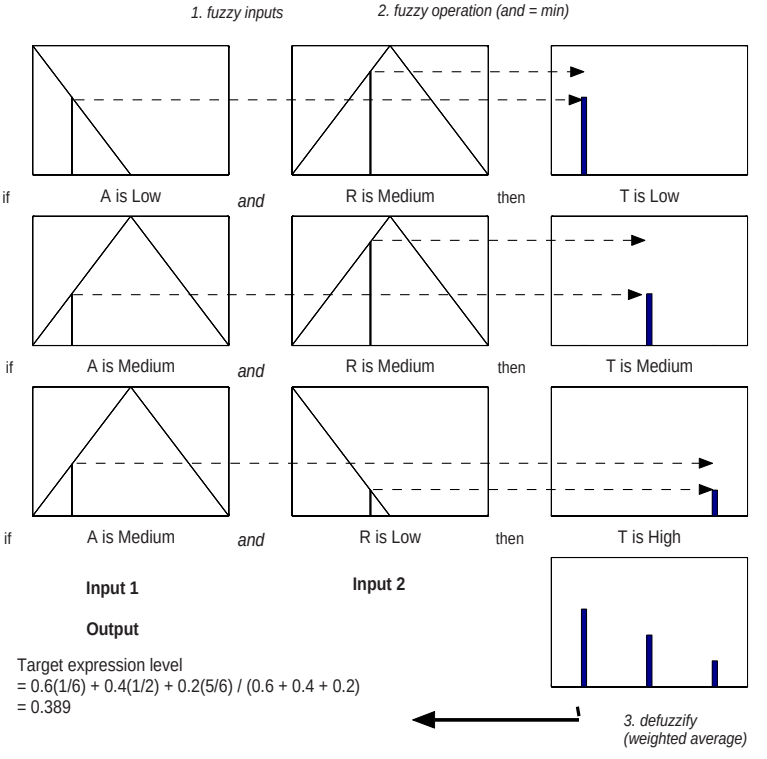


Fig. 8.4. The de-fuzzification process

$G - 2$ genes are considered as potential target genes, and the predicted target expression profile for this gene pair is compared with the actual profiles of these remaining genes. To determine which gene triplets best fit the regulatory network model, each activator / repressor / target gene triplet is evaluated based on a composite score consisting of a residual and variance score. The residual score indicates how accurately the activator / repressor gene pair predicts the target gene expression profile. The variance score measures the variation in configurations the activator / repressor gene pair exhibit over the time course. This is important, because activator and repressor gene pairs which predict well in a variety of situations should be given greater weight. Conversely, if the gene pair only made predictions within a small segment of the decision matrix, the performance of the gene pair for the rest of the decision matrix would be entirely unknown. A low variance score indicates that the activator and repressor pair cover the decision matrix well. An overall score is obtained by multiplying the residual and variance scores together, so that networks with lower overall scores rank higher and exhibit a better fit with the regulatory model.

The methodology presented in [7] extends the three-state fuzzy logic model to an arbitrary number of odd states. The membership functions and decision matrix are both straightforward generalizations of their three-state counterparts. In addition, [7] develop an alternative probability model, which is similar in spirit to the fuzzy logic model. The authors evaluated the number of states in the model using both real and simulated microarray data, and found that a moderate number of states (5 to 7) performed best. The software for running the algorithm is available in the software package `FPRNET` [6]. The user specifies the number of states to use, the number of network triplets to output, and the type of model to run (fuzzy logic or probability). The networks are ranked on the basis of their overall score, and the top scoring triplets are output to a file.

8.3.2 Other Algorithms for Identifying Regulatory Networks Based on Fuzzy Inference

Linear Fuzzy Gene Networks with Accelerated Search

The models developed in [47, 16] combine a linear fuzzy logic model with a genetic search algorithm to identify a set of optimally fitting rule configurations for representing a gene regulatory network. The authors use a three-state membership function, similar to [57], for the fuzzy logic representation of the data. The method is designed for ratiometric data, where the expression ratios are first symmetrized about zero by taking base 2 logarithms. Values are then converted to the range $[-1, 1]$ by taking the arctangent and dividing by $\pi/2$. They are then assigned linguistic values using the three-state membership function.

To avoid the exponential explosion of possible rule combinations that occurs when considering multiple input nodes, the authors restrict the model to a linear fuzzy logic scheme using the union rule configuration of [15]. Under this setup, the rule set complexity grows linearly with the number of input nodes. The effect of each input node on the output is evaluated on an individual basis,

using IF-THEN constructs. For example, one possibility relating the expression of input gene A to output gene B is "If A is Low then B is Low", "If A is Medium then B is Medium", and "If A is High then B is Medium". In [16], the set of the $3^3 = 27$ possible rule combinations is restricted to seven, three *positive* rules (higher expression of the input node leads to higher expression of the output node), three *negative* rules (higher expression of the input node leads to lower expression of the output node), and a null rule corresponding to no effect. The set of possible rules corresponding to a particular gene can be constrained beforehand, which allows for incorporation of prior knowledge concerning the effect of that gene. The number of possible inputs to an output node is not restricted in [16].

To relate the overall effect of G input genes on the output node or target gene, the contribution of all the inputs are aggregated by a fuzzy union, corresponding to a logical OR operation. This is computed by summing the contributions of each input gene to each output state. For an input gene vector $\mathbf{y}$ having fuzzy representation $[\gamma_1, \gamma_2, \gamma_3]$ and fuzzy rule $\mathbf{r} = [r_1, r_2, r_3]$, the contribution to the effect on the output node $\mathbf{z}$ is $[\gamma_{r_1}, \gamma_{r_2}, \gamma_{r_3}]$. For G input genes, the effect of each gene can be represented as an intermediate output $z^i = [z_1^i, z_2^i, z_3^i]$, and the resulting fuzzy value obtained by summing the contributions of each input gene:

$$\mathbf{z} = \sum_G z^i = [\sum_G z_1^i, \sum_G z_2^i, \sum_G z_3^i] \,.$$

The fuzzy representation of the output vector is defuzzified by taking a weighted average of the point masses at -1, 0, and 1, using the fuzzy membership values in $\mathbf{z}$ as weights.

To search the allowable space of fuzzy rule sets, [16] use an evolutionary search algorithm to modify and adapt the rule sets. At each iteration of the algorithm, the overall fitness of the rule sets in the current population should increase. The search algorithm proceeds as follows. At the outset, an intial population of N rule combinations is generated at random, where each rule combination specifies how the G nodes (genes) affect the output node. Each of these N rule combinations is duplicated (the "new" population), and each of the duplicated rule combinations has a specified probability p_C of a crossover event and p_M of a mutation event. In a crossover event, a random number of corresponding rules in two rule combinations are swapped (the other rule combination is randomly selected from the new population). In a mutation event, the rules corresponding to a uniformly distributed random number of input nodes are altered (in accordance with the user specified contraints for those nodes). The fitness function of each rule set in the old and new populations ($2N$ rule sets in total) is evaluated, and the top N rule sets are selected to form the next generation. The fitness function E is the sum of squared errors between the predicted target expression profile and the actual expression profile of the output node (gene), normalized by the variability of the output node to allow comparisons between genes with high and low expression ratio magnitudes. A gamma distribution is used to model the distribution of the error terms E, to determine whether a particular rule set is fitting the output node better than what is expected by chance. The

parameters of the gamma distribution are determined by calculating the error terms corresponding to a suitable number of randomly selected rule sets.

Recurrent Neural Fuzzy Networks

Maraziotis *et. al* proposed a method for modeling the regulation of gene expression based on neural fuzzy recurrent networks (NFRNs) [36]. The NFRN combines the computational power and flexibility of the neural network with the human-like reasoning of fuzzy systems. The recurrent structure of the model allows for dynamic mapping of the expression data, which has advantages over static mapping when time-series expression data are considered (e.g., it allows the incorporation of feedback loops). The model of [36] is based largely on those proposed in [29] and [37].

One advantage of the NFRN model developed by [36] is that both the structure and the parameters of the model are learned. Hence the rules specifying the network architecture are not designated in advance, but are determined by the data. In order to avoid the problem of rule set combinatorial explosion, a clustering-based partition method is used to constrain the number of possible rules describing the causal relationships. As in [7], they find that the number of fuzzy sets needed to adequately model the network can be limited to an intermediate number (seven).

The NFRN architectural structure of [36] consists of six layers (see Figure 8.5). The nodes in the first layer are the input variables, whose values are transmitted directly to the second layer. The second layer performs the fuzzification of the data using Gaussian membership functions, where the number of

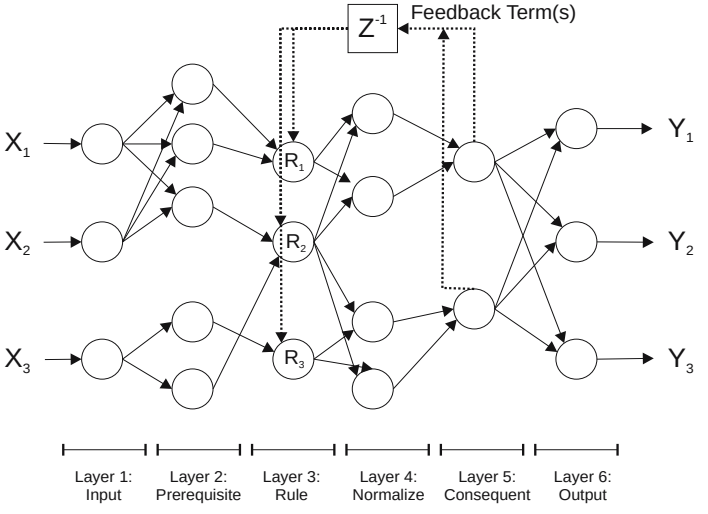


Fig. 8.5. Schematic diagram representing the neural fuzzy recurrent network of Maraziotis *et. al* [36]

linguistic labels is not pre-determined but learned by the algorithm. The nodes in the third layer constitute the rule nodes. The output of each node in this layer depends on input from two sources, layer 2 (the spatial firing degree) and layer 5 (the temporal firing degree). The fourth layer normalizes the output from the third layer, and the fifth layer defuzzifies the data using the output linguistic variables.

The NFRN learns the structure (clustered partitioning) of the input-output space using two criterion, the residual error between the predicted and actual target expression profile and the distance between the proposed rule (added cluster) and all the previously created rules (clusters). The algorithm also checks at the completion of the structure learning phase to see if redundant rules can be deleted or merged with other rules, to create a more parsimonious model and eliminate redundancy. The parameters of the NFRN are then tuned using the back-propagation through time (BPTT) algorithm+[29].

8.3.3 Other Algorithms for Network Reconstruction

In this section, we very briefly mention a few commonly used methods for inference of regulatory networks which do not use a fuzzy logic framework for inference. These methods are compared with the fuzzy logic methods of [7] and [47, 16] using simulated data in Section 8.4. References for each are given for the interested reader.

Bayesian Networks

Bayesian networks (BNs) consist of a graphical structure combined with a family of conditional probability distributions that together define the joint distribution over the set of nodes. Model structures M are sampled from a posterior distribution $P(M|D) \propto P(D|M)P(M)$, where $P(D|M)$ is the likelihood of the data given the model structure M and depends on the model parameters. Sampling from the posterior distribution is achieved via Markov Chain Monte Carlo (MCMC) methods, using an efficient proposal algorithm based on node orders [20]. Numerous extensions to the BN have been developed, including dynamic BNs for modeling time series data and allowing cyclic regulation [27], using a non-parametric regression model to avoid discretisation [26], and incorporation of biological prior knowledge such as transcription factor binding locations [55].

Relevance Networks

Relevance networks (RNs) [9, 10] are a straightforward approach based on pairwise association scores between all pairs of nodes. The association score can be based on the mutual information or the Pearson correlation between the signals associated with each node. The approach has the advantage of being computationally simple, but may have difficulty in distinguishing between direct and indirect interactions (see Figure 8.2).

Graphical Gaussian Models

Graphical Gaussian models (GGMs) [44, 45] are inferred from the matrix of partial correlation coefficients between each node. The partial correlation coefficients are calculated using the inverse of the empirical covariance matrix, which for genomic data can be unstable due to the high-dimensionality of the data. The authors stabilize the estimate of the covariance matrix using a novel regularization approach based on shrinkage [45]. Edges between nodes in the graph are determined by large partial correlation coefficients, and significantly small values are removed from the graph using an empirical Bayes procedure to estimate the local false discovery rate (FDR).

Partial Least Squares

Partial least squares (PLS) was recently proposed by Pihur *et. al* [39]. For each gene, a separate PLS regression model is constructed, using a specified number of PLS components (typically, 2 to 5). The association between the 'dependent' gene (the dependent variable in the PLS regression model) and the remaining genes used to construct the PLS components can be determined by the PLS component coefficients for each gene and the estimated beta parameters in the regression model (essentially, the scalar product of these two vectors for each gene). For undirected edges, the overall association between any two genes is an average of the association measures using each gene as the dependent variable. Similarly to GGMs, this measure of association is adjusted for the effects of other genes, since the PLS components are constructed in an orthogonal manner. An empirical Bayes approach is also used to estimate the local FDR and determine significant edges.

8.4 Simulated Data: Raf Signaling Pathway

We evaluated the fuzzy logic (FL) models of [7] and [16] on previously generated data concerning the Raf signaling pathway, taken from Werhli *et al.* [54]. Raf is a critical signalling protein involved in regulating proliferation of cells in the human immune system, and deregulation of the Raf pathway can lead to carcinogenesis. Figure 8.6 shows the currently accepted signaling network, taken from Sachs *et al.* [43]. We present results from a representative subset of the data sets used in [54]; the observed and interventional real cytoflow data, observed and interventional Gaussian data, and observed data generated by Netbuilder with a noise level $\sigma = 0.3$, under both the correct and V-structure topology. In each situation, 100 observations were generated for each of the 11 proteins, and the process was repeated five times to generate five independent data sets of each type.

The ranking of the network triplets produced by [7] defines a reciever operator characteristic (ROC) curve of the number of true positive (TP) versus false positive (FP) edges. Based on this, two scores were used to compare the different network reconstruction algorithms [54]. The first score records the number

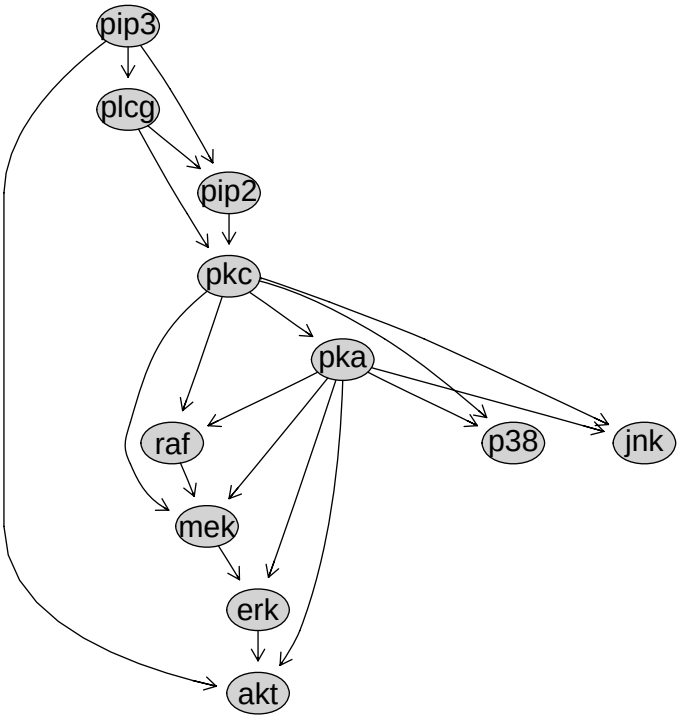


Fig. 8.6. The Raf signalling pathway. Nodes represent proteins, and edges indicate the direction of signal transduction.

of TP edges discovered when the number of FP edges was five. This score captures the performance of each method at low FP values. The second score is the area underneath the ROC curve (AUC), which gives an aggregate measure of performance over all FP values. In both cases, larger values indicate better performance. Each score was calculated under a directed (DGE) and an undirected (UGE) criterion. The DGE considers the directionality of the edge when scoring a correctly specified edge, and the UGE considers only the edge itself.

Tables 8.3 and 8.4 present a subset of the results from [7], for the 5-state FL model. Also added are results using the PLS model of [39]; the results for the relevance network, graphical Gaussian model, and Bayesian network are taken from [54]. The results of the FL model are comparable to RNs, GGMs, and PLS for the DGE TP counts. For the DGE AUC score, the FL model has results comparable to RNs, but is outperformed by the other three methods. The FL model has performance similar to RNs for the UGE scores (TP and AUC) on the real and Gaussian data, but does worse on the Netbuilder data. As noted in [7], however, the intended purpose of the FL model as a screening tool of large genomic data sets is different from the other methods, which are primarily intended for refined modeling of smaller gene sets.

Table 8.3. True positive edges found with no more than five false positive edges, using data generated from the Raf signalling pathway

		Gaussian	Gaussian	Netbuilder	Netbuilder	Real	Real
Measure	Model	Int	Obs	Obs Orig	Obs V-str	Int	Obs
DGE	BN	18.4	4.9	4.1	7.7	6.9	3.3
	GGM	5.2	4.7	4.7	5.5	4.1	5.1
	RN	1.8	3.8	5.1	5.0	1.7	5.1
	PLS	4.4	3.2	4.6	5.8	3.4	5.0
	FL 5	3.0	3.8	3.0	5.6	1.4	3.0
UGE	BN	18.5	15.8	15.5	14.2	11.1	9.5
	GGM	13.2	14.8	14.8	13.2	9.6	9.6
	RN	6.5	8.1	16.6	13.6	7.1	9.3
	PLS	11.6	11.4	12.8	15.0	10.6	9.8
	FL 5	6.6	8.0	4.8	8.0	6.2	6.6

Table 8.4. AUC values for the ROC curve of true postive versus false positive edges found, using data generated from the Raf signalling pathway

		Gaussian	Gaussian	Netbuilder	Netbuilder	Real	Real
Measure	Model	Int	Obs	Obs Orig	Obs V-str	Int	Obs
DGE	BN	0.980	0.782	0.821	0.875	0.697	0.623
	GGM	0.749	0.797	0.798	0.835	0.666	0.644
	RN	0.663	0.641	0.824	0.845	0.553	0.631
	PLS	0.797	0.819	0.832	0.843	0.695	0.679
	FL 5	0.709	0.614	0.641	0.759	0.555	0.578
UGE	BN	0.966	0.885	0.905	0.933	0.791	0.690
	GGM	0.820	0.881	0.883	0.904	0.713	0.685
	RN	0.710	0.681	0.916	0.915	0.569	0.668
	PLS	0.718	0.727	0.774	0.779	0.662	0.625
	FL 5	0.694	0.656	0.535	0.708	0.571	0.639

Table 8.5 gives results from analyzing the Raf signalling pathway data using the FL model of [16]. The networks were estimated using a genetic search algorithm with a population size of 30, 30 generations, and crossover and mutation probabilities of 0.7. The algorithm was run with five different starting seeds, and the best rules for each output gene after all iterations were retained. All possible regulatory connections were allowed except for auto-regulation. Since the model produces one final network indicating the type of interaction between genes, but no relative ranking of the edges, we could not calculate the TP and AUC scores used in [54]. Instead, we present the TP, FP, sensitivity, and specificity of the estimated network models. The number of TP edges for each data set are higher than those in Table 8.3, but so are the number of FP edges, making direct comparisons with the other methods difficult. Of note, the sensitivity and specificity for the DGE and UGE measures are fairly comparable, indicating that the model does well at detecting the directionality of the edge in addition to the presence

Table 8.5. True positive (TP), false positive (FP), sensitivity, and specificity for the fuzzy logic model of [16], using data generated from the Raf signalling pathway

Measure	Data Set	TP	FP	Sensitivity	Specificity
UGE	Gaussian Int	28.0	20.4	0.70	0.75
	Gaussian Obs	30.4	34.8	0.76	0.57
	Real Int	31.6	34.4	0.79	0.58
	Real Obs	28.8	36.4	0.72	0.55
	Netbuilder Obs Orig	37.2	38.8	0.93	0.52
	Netbuilder Obs V-str	35.2	40.4	0.88	0.50
DGE	Gaussian Int	12.2	26.8	0.61	0.73
	Gaussian Obs	12.8	37.2	0.64	0.63
	Real Int	13.6	41.2	0.68	0.59
	Real Obs	10.6	36.2	0.53	0.64
	Netbuilder Obs Orig	17.8	45.4	0.89	0.55
	Netbuilder Obs V-str	17.0	44.4	0.85	0.56

of the edge. We also ran the model allowing for all possible regulatory connections, including auto-regulation. The resulting networks were more sparse and had higher specificity values, but also dropped in sensitivity (results not shown).

8.5 Yeast Cell-Cycle Data

We also tested the FL models of [7] and [16] on gene expression data from the yeast cell-cycle [48]. We used the data from the *alpha*, *cdc15*, *cdc28*, and *elu* cell-synchronized data sets, with 18, 24, 17, and 14 time points, respectively. Missing values were imputed using the K-Nearest-Neighbor (KNN) algorithm [8]. The data sets were analyzied in two ways. First, we screened the data in a manner similar to [57] and [7], to eliminate genes which had expression levels below the noise threshold and did not fluctuate significantly during the cell cycle. In particular, all genes with a maximum expression level below the first quartile in any of the four data sets or exhibited less than a three fold difference between maximum and minimum expression values in all four data sets were removed. The resulting data set of 1737 genes was screened using the FL model of [7], and the gene triplets were assessed for enrichment of transcription factors and cell-cycle regulated genes. In the second analysis, we selected twelve genes from the yeast cell-cycle pathway taken from KEGG, given in Table 8.6. We ran the FL genetic search algorithm model of [16] on the gene expression values corresponding to these twelve genes, and checked for consistency of the estimated network with the pathway depicted in KEGG.

Screening of Network Triplets

To assess whether the FL model of [7] is returning biologically plausible results, we examined the number of network triplets that contained genes annotated as transcription factors in activator and repressor positions, and cell-cycle regulated

Table 8.6. Subset of genes selected for constructing regulatory networks using the FL model of [16]

Gene Name	ORF
CLN3	YAL040C
CDC28	YBR160W
MBP1	YDL056W
SWI4	YER111C
CDC20	YGL116W
CLB6	YGR109C
CDC6	YJL194W
SIC1	YLR079W
SWI6	YLR182W
CLN1	YMR199W
CLN2	YPL256C
CLB5	YPR120C

genes in the target position. Brock *et al.* [7] presented results for the cell-cycle data of [14]. Here, we extend the analysis to include three additional data sets from [48] (*alpha*, *cdc15*, and *elu*). We obtained two lists of genes affiliated with the regulation of the transcription process, both using the *YEAST* package [33] from the Bioconductor repository [23]. The first was obtained from all genes containing the phrase "transcription factor" in their description, resulting in 116 genes. The second, larger list was obtained by all genes containing the word "transcription" in their description. For cell-cycle regulated genes, we used three lists of increasing size. The first list contained 104 genes known to be regulated at various phases of the cell-cycle [48]. The second list contained 421 genes determined to be cell-cycle regulated by Cho *et al.* [14]. The third list contained 800 genes determined to be cell-cycle regulated by Spellman *et al.* [48].

The percentage of the top scoring networks containing transcription factors in the activator / repressor positions and cell-cycle regulated genes in the target positions are given in Table 8.7, for both a 7 state and 15 state FL model. The enrichment of cell-cycle regulated genes in target positions is striking, with between a 2 to 5 fold increase in representation among the top scoring networks compared to the nominal frequency of occurrence in the reduced data. For transcription factors the enrichment is not so pronounced, with between a 1.5 to 2 fold increase in representation for the 7 state model, but little if any enrichment for the 15 state model. The lack of over-representation of transcription factors in the top scoring networks may be explained by incongruency between mRNA transcript levels and transcription factor activity. For example, a regulatory protein may be maintained at a fairly constant level within the cell, and activated or deactivated (e.g. via phosphorylation) when necessary. Thus the amount of mRNA transcript present may not reflect the activity of gene's protein within the cell.

Table 8.7. Percent of the top scoring networks with transcription factors in the activator and repressor positions, and cell-cycle regulated genes in the target position. The different lists of transcription factor (TF) and cell-cycle regulated (CC) genes are described in the text.

		Number in Reduced		Top 10,000 (%)		Top 1,000 (%)	
List	Number	Data (%)	Position	7 State	15 State	7 State	15 State
TF1	116	27 (1.7)	Act	3.4	1.9	2.8	1.9
			Rep	3.1	1.6	2.2	1.6
TF2	530	109 (7.1)	Act	10.8	7.1	8.0	7.1
			Rep	10.0	8.7	11.3	8.7
CC1	104	68 (4.4)	Tar	23.4	19.4	25.3	19.4
CC2	421	226 (14.6)	Tar	62.3	53.5	73.1	53.5
CC3	800	449 (29)	Tar	80.3	69.8	90.3	69.8

KEGG Cell-Cycle Pathway

We ran the FL model of [16] using the *alpha*, *cdc15*, *cdc28*, and *elu* yeast cell-cycle data from [48] on the twelve genes in Table 8.6. The model was run with the same parameter settings as in Section 8.4 and assuming no auto-regulation, but all other regulatory connections were allowed. The final predicted network is given in Table 8.8. The results presented here have been simplified by indicating all positive interactions with a 1, all negative interactions with a -1, and no interaction with a 0.

We evaluated the results by comparing with the cell-cycle figure as depicted in KEGG [24] (see Figure 8.1), with connections assumed to exist between genes

Table 8.8. Predicted regulatory network using the FL model of [16], for the twelve cell-cycle genes in Table 8.6. A 1 indicates a positive interaction, a -1 negative interaction, and 0 no interaction. The row genes indicate the origin of the edge (the regulatory genes), the column genes are the targets. Correctly specified edges (direction and type) are given in bold.

	CLN3	CDC28	MBP1	SWI4	CDC20	CLB6	CDC6	SIC1	SWI6	CLN1	CLN2	CLB5
CLN3	0	1	**1**	−1	0	0	1	0	0	0	0	1
CDC28	**1**	0	0	0	1	**1**	0	−1	**1**	0	0	0
MBP1	0	**1**	0	−1	0	0	0	−1	−1	0	−1	0
SWI4	−1	0	−1	0	0	1	1	1	0	0	0	1
CDC20	0	**1**	0	0	0	0	1	1	1	0	−1	−1
CLB6	1	**1**	1	1	−1	0	0	1	−1	1	1	**1**
CDC6	1	1	−1	1	1	0	0	1	−1	0	0	0
SIC1	0	**−1**	−1	1	1	0	1	0	0	0	0	0
SWI6	−1	**1**	−1	0	0	0	0	0	0	**1**	0	0
CLN1	0	**1**	1	0	−1	1	0	0	1	0	**1**	1
CLN2	−1	**1**	−1	1	−1	1	0	0	1	1	0	1
CLB5	0	0	0	1	−1	**1**	**1**	0	0	0	1	0

within a complex. When considering both correct specification of edge direction and type (positive vs negative regulation), the algorithm correctly identified 36.4% of the true associations, with a specificity of 47% and an overall accuracy of 43.8%. Compared to results reported in other cases [39], the algorithm achieves respectable success. Also, given that only three of the interactions were considered negative in the true network, it is difficult for any algorithm to correctly specify the type of interaction without some prior constraints. When considering just the directionality of the edge, the sensitivity increased to 47.7% and the overall accuracy to 47.2%, while the specificity remained unchanged. Since the model will always try to best explain a given gene's expression pattern, correctly characterizing a protein which is solely regulatory within a system (e.g., CDC20 in our example) is a difficult task. Knowledge of a whether a connection exists, though, is still useful to biologists, who can design further assays to study the nature of the relationship between the genes. Hence the ability of an algorithm to correctly specificy an existing edge, irrespective of direction, is important. In this case, the FL model captures 71.6% of the edges between genes, with a specificity of 37.7% and an overall accuracy of 53.5%.

8.6 Conclusion

In this chapter, we have demonstrated the use of fuzzy logic to develop tools capable of screening genomic data sets for potential regulatory interactions, and comprehensive reverse engineering of regulatory pathways concerning a subset of genes of interest. In both cases, the use of fuzzy logic allows the incorporation of high-level, human-like reasoning in constructing the rule-base for the regulatory system. This is especially advantageous for the biologist or subject matter expert who has extensive knowledge about existing regulatory mechanisms, who can then cast this information in an intuitive fashion using the semantics of fuzzy logic.

Fuzzy logic models are flexible, and several aspects can be modified to possibly achieve greater performance. For example, alternative inference systems (e.g., Mamdani [35, 34] and Tsukamoto [50]), and input-output membership function shape and spacing can be used. In addition, non-linear effects among the regulatory genes can be added by consulting with subject matter experts and translating knowledge about gene interactions into an appropriate fuzzy logic model. The FL model of [16] learns the rule configuration using a genetic algorithm, while the NFRN of [36] fine-tunes the parameters of the regulatory system using the BPTT algorithm. However, learning these aspects of the FL system adds computational complexity to the algorithm. If the method is primarily intended as an exploratory tool, as in [7], then the increase in computational overhead must be weighed against the added flexibility of the model. Both the [7] and [16] models were assessed using simulated microarray data, and results compared favorably with other network reconstruction algorithms. Unfortunately, the NFRN program was not publicly available at the time of writing, and so was not included in the comparative analysis. Results from analyzing

publicly available yeast cell-cycle data [48] indicate the FL models also return biologically meaningful results.

There are inherent difficulties in discovering regulatory networks using microarray data. Gene expression is regulated at several stages, including DNA transcription, RNA processing and transport (in eukaryotes), RNA translation, and post-translational modification of proteins. DNA microarrays capture biological activity involving the first of these stages, but regulation at later stages is not necessarily reflected by changes in mRNA transcript abundance. Further, interactions gleaned from microarray data are not necessarily causative in nature, but may indicate indirect connections among genes involved in similar biological pathways. Further experimental assays are needed to determine the validity of genetic interactions suggested by network models. In spite of these challenges, the computational approaches discussed here provide useful tools to mine microarray data for potential genetic regulatory interactions. This information can generate testable hypothesis and guide future experiments, focusing the efforts of investigators and saving them time and resources.

References

1. Akutsu, T., Miyano, S., Kuhara, S.: Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In: Pacific Symposium on Biocomputing, vol. 99(4), pp. 17–28 (1999)
2. Azuaje, F.: A computational neural approach to support the discovery of gene function and classes of cancer. IEEE Trans. Biomed. Eng. 48(3), 332–339 (2001)
3. Berggard, T., Linse, S., James, P.: Methods for the detection and analysis of protein-protein interactions. Proteomics 7(16), 2833–2842 (2007)
4. Bolstad, B.M., Irizarry, R.A., Astrand, M., Speed, T.P.: A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. Bioinformatics 19(2), 185–193 (2003)
5. Bork, P., Copley, R.: The draft sequences. filling in the gaps. Nature 409, 818–820 (2001)
6. Brock, G.N.: FPRNET: Fuzzy logic, probability, and regression models for network reconstruction, Version 1.0 (2008)
7. Brock, G.N., Beavis, W.D., Kubatko, L.S.: Fuzzy logic and related methods as a screening tool for detecting gene regulatory networks, Information Fusion (in press)
8. Brock, G.N., Shaffer, J.R., Blakesley, R.E., Lotz, M.J., Tseng, G.C.: Which missing value imputation method to use in expression profiles: a comparative study and two selection schemes. BMC Bioinformatics 9, 12 (2008)
9. Butte, A.J., Kohane, I.S.: Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. In: Pacific Symposium on Biocomputing, pp. 418–29 (2000)
10. Butte, A.J., Tamayo, P., Slonim, D., Golub, T.R., Kohane, I.S.: Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. Proceedings of the National Academy of Science, USA 97(22), 12182–12186 (2000)
11. Chaves, M., Sontag, E.D., Albert, R.: Methods of robustness analysis for boolean models of gene control networks. Systems Biology (Stevenage) 153(4), 154–167 (2006)

12. Chen, C.F., Feng, X., Szeto, J.: Identification of critical genes in microarray experiments by a neuro-fuzzy approach. Comput. Biol. Chem. 30(5), 372–381 (2006)
13. Chen, T., He, H.L., Church, G.M.: Modeling gene expression with differential equations. In: Pacific Symposium on Biocomputing, vol. 99(4), pp. 29–40 (1999)
14. Cho, R.J., Campbell, M., Winzeler, E., Steinmetz, L., Conway, A., Wodicka, L., Wolfsberg, T., Gabrielian, A., Landsman, D., Lockhart, D., Davis, R.: A genome-wide transcriptional analysis of the mitotic cell cycle. Molecular Cell 2, 65–73 (1998)
15. Combs, W.E., Andrews, J.E.: Combinatorial rule explosion eliminated by a fuzzy rule configuration. IEEE Transactions on Fuzzy Systems 6(1), 1–11 (1998)
16. Datta, S., Sokhansanj, B.A.: Accelerated search for biomolecular network models to interpret high-throughput experimental data. BMC Bioinformatics 8, 258 (2007)
17. D'Haeseleer, P., Wen, X., Fuhrman, S., Somogyi, R.: Linear modeling of mRNA expression levels during CNS development and injury. In: Pacific Symposium on Biocomputing, vol. 99(4), pp. 41–52 (1999)
18. Dudoit, S., Yang, Y.H., Callow, M.J., Speed, T.P.: Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. Statistical Sinica 12(1), 111–139 (2002)
19. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Science USA 95, 14863–14868 (1998)
20. Friedman, N., Koller, D.: Being Bayesian about network structure. Machine Learning 50, 95–126 (2003)
21. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian networks to analyze expression data. Journal of Computational Biology 7, 601–620 (2000)
22. Futschik, M.E., Chaurasia, G., Herzel, H.: Comparison of human protein-protein interaction maps. Bioinformatics 23(5), 605–611 (2007)
23. Gentleman, R.C., Carey, V.J., Bates, D.M., Bolstad, B., Dettling, M., Dudoit, S., Ellis, B., Gautier, L., Ge, Y., Gentry, J., Hornik, K., Hothorn, T., Huber, W., Iacus, S., Irizarry, R., Leisch, F., Li, C., Maechler, M., Rossini, A.J., Sawitzki, G., Smith, C., Smyth, G., Tierney, L., Yang, J.Y.H., Zhang, J.: Bioconductor: Open software development for computational biology and bioinformatics. Genome Biology 5, R80 (2004)
24. Kanehisa, M., Goto, S., Kawashima, S., Nakaya, A.: The KEGG databases at genomenet. Nucleic Acids Res 30(1), 42–46 (2002)
25. Kerr, M.K., Churchill, G.A.: Experimental design for gene expression microarrays. Biostatistics 2, 183–202 (2001)
26. Kim, S., Imoto, S., Miyano, S.: Dynamic bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data. Biosystems 75(1-3), 57–65 (2004)
27. Kim, S.Y., Imoto, S., Miyano, S.: Inferring gene networks from time series microarray data using dynamic Bayesian networks. Briefings in Bioinformatics 4, 228–235 (2003)
28. Latchman, D.S.: Eukaryotic transcription factors, 4th edn. Academic Press, London (2003)
29. Lee, C.H., Teng, C.C.: Identification and control of dynamic systems using recurrent fuzzy neural networks. IEEE Transactions on Fuzzy Systems 8(4), 349–366 (2000)
30. Lee, M.L., Bulyk, M.L., Whitmore, G.A., Church, G.M.: A statistical model for investigating binding probabilities of DNA nucleotide sequences using microarrays. Biometrics 58, 981–988 (2002)

31. Liang, S., Fuhrman, S., Somogyi, R.: REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. In: Pacific Symposium on Biocomputing, vol. 98(3), pp. 18–29 (1998)
32. Linden, R., Bhaya, A.: Evolving fuzzy rules to model gene expression. Biosystems 88(1-2), 76–91 (2007)
33. Liu, T.Y., Lin, C.W., Falcon, S., Zhang, J., MacDonald, J.W.: Yeast: A data package containing annotation data for yeast, R package version 2.0.1 (2008)
34. Mamdani, E.H.: Applications of fuzzy logic to approximate reasoning using linguistic synthesis. IEEE Transactions on Computers 26(12), 1182–1191 (1977)
35. Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies 7(1), 1–13 (1975)
36. Maraziotis, I.A., Dragomir, A., Bezerianos, A.: Gene networks reconstruction and time-series prediction from microarray data using recurrent neural fuzzy networks. IET Syst. Biol. 1(1), 41–50 (2007)
37. Mastorocostas, P.A., Theocharis, J.B.: A recurrent fuzzy-neural model for dynamic system identification. IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics 32(2), 176–190 (2002)
38. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science 298(5594), 824–827 (2002)
39. Pihur, V., Datta, S., Datta, S.: Reconstruction of genetic association networks from microarray data: a partial least squares approach. Bioinformatics 24(4), 561–568 (2008)
40. Ping, X., Brock, G.N., Parrish, R.S.: Modified linear discriminant analysis approaches for classification of high-dimensional microarray data. Computational Statistics and Data Analysis (in press)
41. Ren, B., Robert, F., Wyrick, J.J., Aparicio, O., Jennings, E.G., Simon, I., Zeitlinger, J., Schreiber, J., Hannett, N., Kanin, E., Volkert, T.L., Wilson, C.J., Bell, S.P., Young, R.A.: Genome-wide location and function of dna binding proteins. Science 290(5500), 2306–2309 (2000)
42. Ressom, H., Reynolds, R., Varghese, R.S.: Increasing the efficiency of fuzzy logic-based gene expression data analysis. Physiol Genomics 13(2), 107–117 (2003)
43. Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D.A., Nolan, G.P.: Causal protein-signaling networks derived from multiparameter single-cell data. Science 308(5721), 523–529 (2005)
44. Schäfer, J., Strimmer, K.: An empirical bayes approach to inferring large-scale gene association networks. Bioinformatics 21(6), 754–764 (2005)
45. 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, Article32 (2005)
46. Shen-Orr, S.S., Milo, R., Mangan, S., Alon, U.: Network motifs in the transcriptional regulation network of escherichia coli. Nat Genet 31(1), 64–68 (2002)
47. Sokhansanj, B.A., Fitch, J.P., Quong, J.N., Quong, A.A.: Linear fuzzy gene network models obtained from microarray data by exhaustive search. BMC Bioinformatics 5, 108 (2004)
48. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D., Futcher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. Mol. Biol. Cell. 9(12), 3273–3297 (1998)

49. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. IEEE Transactions on Systems, Man, and Cybernetics 15(1), 116–132 (1985)
50. Tsukamoto, Y.: An approach to fuzzy reasoning methods. Advances in Fuzzy Set Theory and Applications, 137–149 (1979)
51. van Someren, E.P., Wessels, L.F., Backer, E., Reinders, M.J.: Genetic network modeling. Pharmacogenomics 3, 507–525 (2002)
52. von Mering, C., Krause, R., Snel, B., Cornell, M., Oliver, S.G., Fields, S., Bork, P.: Comparative assessment of large-scale data sets of protein-protein interactions. Nature 417(6887), 399–403 (2002)
53. Weaver, D.C., Workman, C.T., Stormo, G.D.: Modeling regulatory networks with weight matrices. In: Pacific Symposium on Biocomputing, vol. 99(4), pp. 112–123 (1999)
54. Werhli, A.V., Grzegorczyk, M., Husmeier, D.: Comparative evalution of reverse engineering gene regulatory networks with relevance networks, graphical Gaussian models, and Bayesian networks. Bioinformatics 22, 2523–2531 (2006)
55. Werhli, A.V., Husmeier, D.: Reconstructing gene regulatory networks with bayesian networks by combining expression data with multiple sources of prior knowledge, Statistical Applications in Genetics and Molecular Biology 6, Article15 (2007)
56. Wolfsberg, T., McEntyre, J., Schuler, G.: Guide to the draft human genome. Nature 409, 824–826 (2001)
57. Woolf, P.J., Wang, Y.: A fuzzy logic approach to analyzing gene expression data. Physiological Genomics 3, 9–15 (2000)
58. Xing, B., van der Laan, M.J.: A causal inference approach for constructing transcriptional regulatory networks. Bioinformatics 21(21), 4007–4013 (2005)
59. Zou, M., Conzen, S.D.: A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. Bioinformatics 21(1), 71–79 (2005)

9

Fuzzy System Methods in Modeling Gene Expression and Analyzing Protein Networks

Shihua Zhang[1,2], Rui-Sheng Wang[3,5], Xiang-Sun Zhang[1], and Luonan Chen[4,5]

[1] Academy of Mathematics and Systems Science
Chinese Academy of Sciences, Beijing 100080, China
[2] Graduate University of Chinese Academy of Sciences, Beijing 100049, China
[3] School of Information, Renmin University of China, Beijing 100872, China
[4] Institute of Systems Biology, Shanghai University, Shanghai 200444, China
[5] Department of Electrical Engineering and Electronics, Osaka Sangyo University, Osaka 574-8530, Japan

Summary. Recent technological advances in high-throughput data collection allow for computational study of complex biological systems on the scale of the whole cellular genome and proteome. Gene regulatory network is expected to be one of suitable tools for interpreting the resulting large amount of genomic and proteomic data sets. A huge number of methods have been developed for extracting gene networks from such data. Fuzzy logic which plays an important role in multiple disciplines is a framework bringing together physics-based models with more logical methods to build a foundation for multi-scale bio-molecular network models. Biological relationships in the best-fitting fuzzy gene network models can successfully recover direct and indirect interactions from previous knowledge to result in more biological insights about regulatory and transcriptional mechanism. In this chapter, we survey a class of models based on fuzzy logic with particular applications in reconstructing gene regulatory networks. We also extend our survey of the application of fuzzy logic methods to highly related topics such as protein interaction network analysis and microarray data analysis. We believe that fuzzy logic-based models would take a key step towards providing a framework for integrating, analyzing and modeling complex biological systems.

9.1 Introduction

Fuzzy set, which is a kind of logic using a range of values as 'degree of truth' instead of the binary values 'true or false', was first founded by Zadeh [50] and later investigated by many other researchers [21]. So far fuzzy system methods, especially fuzzy logic methods have gained rapid advances. Fuzzy logic has become a widely used computational tool for formulating and transferring human expert knowledge to quantitative models. It provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information, which makes fuzzy logic flexible for modeling the relationship between input and output information and distinguished by its robustness with respect to noise and variations in system parameters. This characteristic mirrors

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 165–189.
springerlink.com

the robustness of many systems and their remarkable ability to achieve precise functional control from imprecise inputs. For example, a fuzzy logic biochemical model requires some numerical parameters in order to operate, such as initial values and rate coefficients, but exact values of these numbers are usually not critical. Since dynamic rules are defined in terms of fuzzy quantities, fuzzy logic models allow computation of logical consequences of complex system dynamics with imprecise variables. At the same time, fuzzy logic models are capable of representing extremely complex systems to high degrees of accuracy when precise data is available. Most biological data are derived from logically-designed, hypothesis-driven experiments, which may contain various noises, fuzzy logic provides a way for biologists to incorporate data that might otherwise be difficult to incorporate into computer models. Therefore, fuzzy logic has been particularly useful in applications where appropriate mathematical models cannot be derived due to the complexity of the problem and has become ubiquitous in modern control systems engineering, including biological and medical applications [44] such as analyzing and modeling gene expression data [10, 45], reconstructing gene regulatory networks [46], etc. In real applications, statistical and non-gradient based optimization methods such as genetic algorithms are often adopted to determine a set of optimal fuzzy rules describing a systems. Here, we briefly describe some fuzzy system methods used in this chapter.

9.1.1 Fuzzy Logic and Fuzzy Systems

Fuzzy logic, the logic based upon which fuzzy systems operate, is much closer in spirit to human thinking and natural language than conventional digital logic. Basically, it provides an effective means of capturing the approximate and inexact nature of the real-world knowledge. According to Zadeh [50], the essential characteristics of fuzzy logic are:

- Exact reasoning is viewed as a limiting case of approximate reasoning
- Everything is to a matter of degree
- Any logic system can be fuzzified
- Knowledge is interpreted as a collection of elastic or equivalent, fuzzy constraints on a collection of variables
- Inference is viewed as a process of propagation of elastic constraints

Fuzzy sets are qualitative properties (e.g., low, medium, high) whose elements belong to the sets only in a degree. The degree of belonging is defined by the value of a membership function (MF), which has values between 0 and 1. Such a technique clearly provides a way of representing uncertainties in a mathematical model. The most popular membership functions are triangular functions, Gaussian functions, bell-shaped functions, and trapezoidal functions. An illustration for bell-shaped membership functions and triangular membership functions is shown in Fig. 9.1. It is defined as:

$$\mu(x) = \frac{1}{1 + [(\frac{x-1}{a})^2]^b}$$

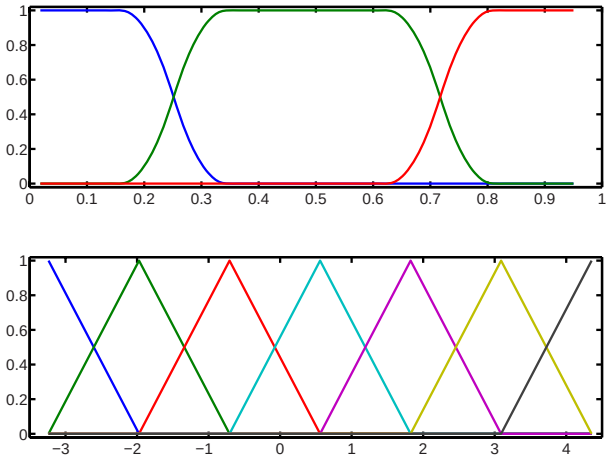


Fig. 9.1. An illustration of bell-shaped membership functions and triangular membership functions

Fuzzy systems are fuzzy rule-based expert systems, which comprise a set of fuzzy rules also known as linguistic rules in the form of 'IF-THEN' [22]. A fuzzy rule is defined as IF x THEN y, where x (condition or premise) is a conjunction in which each clause specifies an input variable and one of the membership functions associated with it, and y (conclusion or consequence) specifies an output variable membership function. Both x and y are propositions containing linguistic variables. It is such linguistic variables and fuzzy IF-THEN rules that enable fuzzy system methods to exploit the imprecisions and uncertainty underlying target data. Fuzzy systems can be broadly categorized into two families. The first includes linguistic models based on collections of IF-THEN rules, whose antecedents and consequents utilize fuzzy values. This family of fuzzy systems uses fuzzy reasoning and the system behavior can be described in natural terms. The second category, based on Sugeno-type systems, uses a rule structure that has fuzzy antecedent and functional consequent parts. Fuzzy rule-based systems have particularly successful applications in computer control and engineering, now extended to other fields such biomedical diagnosis, modeling gene regulation, etc.

9.1.2 Fuzzy Clustering

Clustering of numerical data, which is a branch of pattern recongnition, forms the basis of many classification and system modeling algorithms. The purpose of clustering is to identify natural groupings of data from a large data set to produce a concise representation of a system's behavior. The resulting partition can improve our understanding for the data and reveal the internal structure of the data. While classical hard clustering methods such as k-means, hierarchical clustering have played important roles in analyzing large-scale complex data, in

real applications, there may not be sharp boundaries between different clusters. Fuzzy c-means (FCM) is a popular data clustering technique which allows each data point to belong to a cluster to some degree specified by a membership grade. This technique was originally introduced by Bezdek [4] as a fuzzy-logical extension to earlier clustering methods. It provides a method of how to group data points that populate some multi-dimensional space into a specific number of different clusters. In the recent years, many variants of FCM such as fuzzy j-means have been proposed for more proper applications [3].

9.2 Basic Biological Knowledge

In this section, we briefly introduce some biological concepts and knowledge used in the later sections.

9.2.1 Gene Expression Principle and Data

Although genome stores biological information, it is unable to release the information by itself to the cell. Only through the coordinated activity of enzymes and other proteins participating in a series of biochemical reactions, biological information can be utilized and biological function can be achived. Regulation of gene expression is one of the most important processes in a cell system. It transmits static information encoded in the DNA sequence into functional protein molecules which in turn control most of the cellular processes. A gene is called expressed when it is transferred to RNA molecules which are finally synthesized into proteins through this process. In other words, gene expression is the process by which the inheritable information that comprises a gene (DNA segment) is made manifest as a physical and biologically functional gene product, such as protein or RNA. Gene regulation and gene expression are achieved by a kind of binding proteins known as transcription factors (TFs) which attach to specific DNA promoter regions and initiate RNA synthesis to achieve the transcription process.

The amount of mRNA synthesized during transcription measures how active a gene is. It is thousands of genes and their products in a given living organism that function in a complicated and coordinated way and create the mystery of life. However, traditional methods in molecular biology that examine gene expression levels were limited to a small scale per experiment. With the development of high-density DNA chip technology, a new technology called DNA microarray which can be used to detect RNAs that may or may not be translated into active proteins, has enabled researchers to monitor the whole genome at the transcriptional level on a single chip and detect mRNA expression levels of thousands of genes simultaneously. This kind of analysis is referred to as expression analysis or expression profiling. The first use of microarrays for gene expression profiling was in [37]. Nowadays, various such high-throughput microarray techniques have generated massive amount of data. There are many databases such as Stanford Microarray Database, Gene Expression Omnibus, ExpressDB. in which various microarray data sets are deposited and can be dealt with and

analyzed by different computational methods and help us to gain insights into underlying biological processes. The resulting gene expression data can be used to study the effects of drug treatments, diseases, and developmental genetics, etc. For example, microarray-based gene expression profiling has been widely used to identify disease genes by comparing their expression profiles in diseased and normal tissues. In addition, gene expression profiles are actually the results of transcription regulation, thus they have been widely used to reconstruct gene regulatory networks [46].

9.2.2 Gene Regulatory Networks

As mentioned in last subsection, gene expression means the process of producing functional molecules such as RNA or protein from static DNA sequences, among which transcription regulation is the first and important step. The steps in the gene expression process may be modulated, including the post-transcriptional regulation of a mRNA and the post-translational modification of a protein. One of the most important question in biology is how gene expression is switched on and off, i.e., how the expression of a gene is regulated. The transcriptional regulation of genes is achieved by binding proteins that attach to specific DNA promoter regions and exert their effects positively or negatively on binding of RNA polymerase to promoter region of the gene. Such regulatory relationships between genes can be described by a network. Gene regulatory network (GRN) is a logical way of attempting to describe the relationships between different genes observed with transcription profiling. It is a directed network with genes as nodes and the relationships as edges and can be viewed as a input-output device (Fig. 9.2). The interactions in gene regulatory networks may not be a direct physical interaction from a regulator to a target gene since they do not represent explicitly the proteins and metabolites that mediate those interactions. It is such regulatory networks in cells that dynamically orchestrate the expression level of each gene in the genome by controlling whether and how the gene will be transcribed into RNA in response to various environmental and developmental signals.

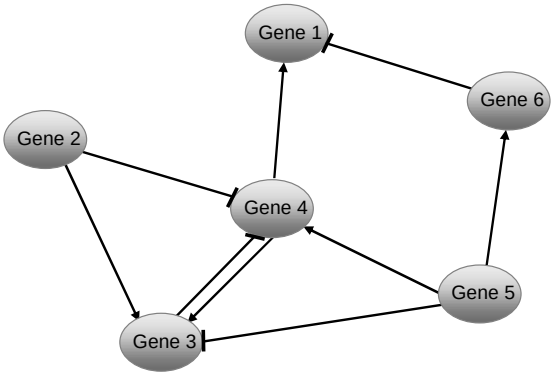


Fig. 9.2. Graphical representation of a gene regulatory network with six genes

Gene regulatory networks can be modeled and simulated by mathematical and computational approaches such as Boolean networks, Bayesian networks, differential equations [2, 9, 18]. Although such models are created based on a set of unrealistic assumptions, they can provide the rough description of system dynamics and derive behavioral predictions. Once the model are chosen, the parameters in the model need to be inferred to fit the data. Since the expression profiles of gene origin from the network interactions between regulators and target genes, it is reasonable to retrieve these interactions from gene expression data. Such a computation process is known as gene regulatory network inference or reverse engineering of gene regulatory networks. In the last few years, a number of methods have been developed for inferring gene regulatory networks from microarray data, among which we will focus on introducing some fuzzy logic methods for modeling gene expression data and reconstructing GRN in this chapter.

9.2.3 Protein Interaction Networks

The nature and role of the interactions between proteins is the central topic in the field of proteomics. Protein-protein interactions play diverse roles in functionality and robustness of biological systems and differ based on the composition, affinity, and lifetime of the association. Non-covalent contacts between residue side-chains which facilitate a variety of interactions and associations within and between proteins are the basis for protein folding, protein assembly, and protein-protein interaction. An interaction may be mainly transient in vivo but becomes permanent under certain cellular conditions. Naturally, proteins seldom act as single isolated units while performing their functions in cellular systems. Proteins involved in the same cellular processes often interact with each other and the function of unknown proteins may be postulated on the basis of their interaction with a known protein target of known function [51]. Mapping protein-protein interactions not only provides insights into protein function but also facilitates the modeling of functional pathways to elucidate the molecular mechanisms of cellular processes. Thus, the study of protein interactions is fundamental to understanding how proteins function together within the cell and how proteins organize to form a robust system.

A huge number of protein-protein interactions have been identified via the yeast two-hybrid system, mass spectrometry, and protein microarrays. Such data have been deposited in several main databases such as DIP, MIPS, etc. However, the data generated is likely to be erroneous and incomplete. In order to form an understanding of the total universe of potential interactions, including those not detected by these methods, it is useful to develop computational methods to predict possible interactions between proteins. The accurate prediction of protein-protein interactions is therefore an important goal in the field of bioinformatics [51].

In protein-protein interaction networks, proteins with tense connections tend to form clusters which correspond to functional modules or protein complexes. Protein complexes are groups of proteins that interact with each other at the

same time and place, forming a single multi-molecular machine. Functional modules consist of proteins that participate in a particular cellular process while binding to each other at a different time and place. Clustering protein-protein interaction networks therefore involves identifying protein complexes and functional modules. It has important significance for analyzing protein interaction networks and their organization, predicting the principal function of each module and elucidating possible protein functions.

Fig. 9.3 shows an overview of the main content of this chapter. Yeast two-hybrid method (Y2H) and Affinity purification combined with mass spectrometry (TAP-MS) are two high-throughput technology for protein interactions. Yeast two-hybrid method (Y2H) is typically carried out by screening a protein of interest against a random library of potential protein partners. Its principle is based on the fact that many eukaryotic transcription activators have at least

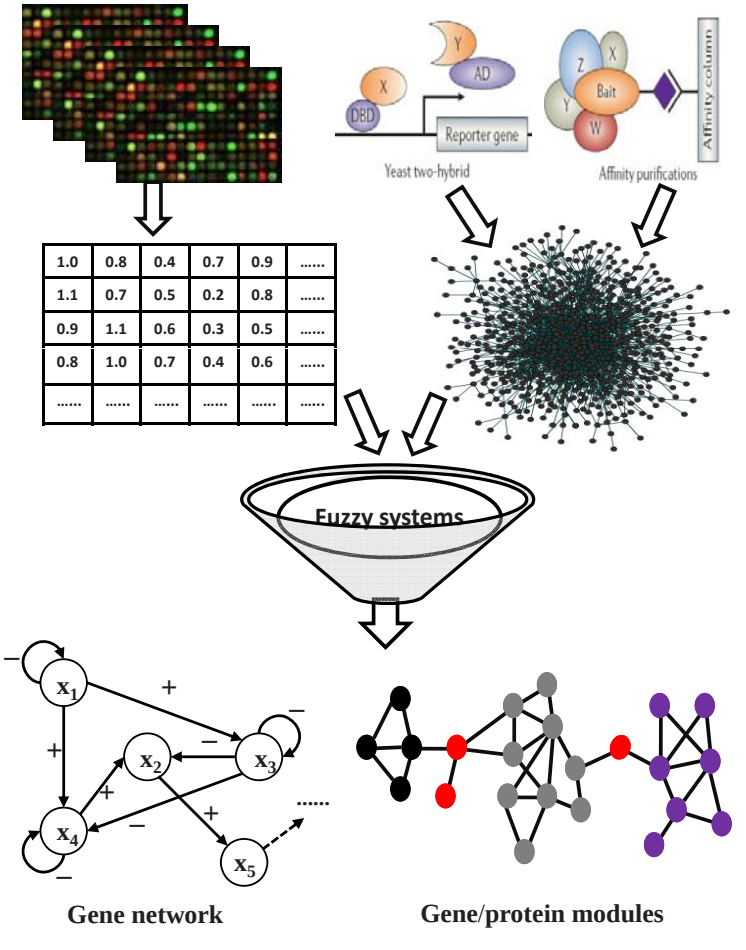


Fig. 9.3. Overview of this chapter

two distinct domains known as DNA-binding do- main (BD) and transcriptional activation domain (AD). In Y2H system, protein-protein interactions are tested by fusing one protein (bait) to the DNA-binding domain of the yeast GAL4 transcription factor. This chimeric protein is cloned in an expression plasmid, which is then transfected into a yeast cell. A similar procedure creates a chimeric sequence of another protein (prey) fused to the GAL4 activation domain. The transcription system works only if the two domains are physically close, which means that splitting BD and AD will inactivate the transcription. But the transcription can be restored if a DNA-binding domain is physically associated with an activating domain. If two proteins physically interact, the reporter gene is activated. Affinity purification combined with mass spectrometry (TAP-MS) approaches typically consist of the selective purification and enrichment of a bait protein and the associated prey proteins that co-purify with the bait. Firstly, appropriate TAP tags are selected. Then, tagged proteins are expressed in yeast and allowed to form physiological complexes. These complexes are affinity purified using the appropriate tag, and the purification protein assemblies are resolved by denaturing gel electrophoresis. Resolved proteins are excised from the gel and then digested by trypsin. The resulting peptides are analyzed by mass spectrometry and interacting proteins can be characterized by using bioinformatics methods.

9.3 Fuzzy System Methods for Reconstructing GRN

As fuzzy system methods have wide applications in other fields such as control theory, decision making, etc, they also have been used in modeling gene expression and reconstructing gene regulatory networks. Fuzzy system methods allow problem solving with incomplete or uncertain information and are suitable for modeling uncertain phenomenon when systems are difficult to describe with a deterministic mathematical model, which makes them an ideal tool for analyzing gene expression data since microarray experimental data are often noisy and uncertain. Therefore, although a large number of computational methods have been developed for building gene regulatory models, fuzzy systems have been effectively used in identifying the logical relationships between genes and are still a promising method to understand the phenomena of gene regulation.

An early work on fuzzy system methods with applications in modeling gene expression was contributed by Woolf and Wang [46], where they built an activator-repressor gene regulatory model based on fuzzy rules to find gene regulation patterns from expression data. Through a fuzzy logic algorithm, gene expression profiles were fuzzified and transformed into qualitative descriptors such as 'high','medium' and 'low'. Then, by using a set of known heuristic fuzzy rules consistent with common knowledge, all possible combinations of triplets (activators, repressors, and target) were tested and the model output was compared to the expression level of the target gene to check if they fit the fuzzy model. Regulation combinations were ranked based upon the residual between the calculated expression and the observed expression of the target gene. Those combinations that have a low error and fit most of the fuzzy rules were inferred to exhibit

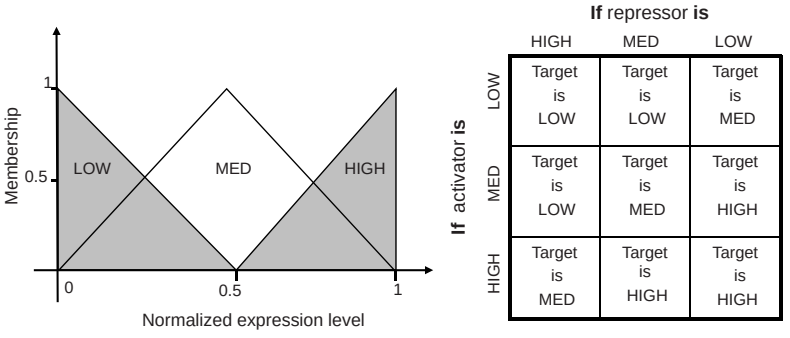


Fig. 9.4. Fuzzy membership functions and fuzzy rules used in [46]

an activator-repressor-target relationship. For example, if the activator is highly expressed and the repressor is lowly expression, the target gene would be highly expressed. If the repressor concentration is high and the activator concentration is low, the target gene would be lowly expressed. The fuzzy membership functions and fuzzy rules used in [46] are shown in Fig. 9.4. Such fuzzy-logic based gene regulatory network model can be viewed as a generalization of Boolean networks since in this model genes have three states instead of only two states 'on' and 'off' in Boolean networks. Furthermore, it performs like a human expert to find underlying regulatory relationships by comparing the expression levels of genes. The predicted regulation patterns constitute a gene regulatory network. This fuzzy-logic model was tested on yeast expression data from *Saccharomyces cervisae* cell cycle expression database and the numerical results agree well with the experimental results from literature.

As pointed out in [35, 34, 33], although above fuzzy-logic model is effective in finding gene regulatory networks, one of the major problems lying in this approach is that large amount of computational time is required for analysis since all possible combinations of triplets should be examined. For example, The analysis of the relationships between 1,898 genes required 200 h on an 8-processor SGI Origin 2000 system. The complexity of the regulatory model increases in $O(n^3)$ as the number of genes n used for the model increases. In addition, the algorithm can only focus on simple regulation patterns and cannot scale well to more complex models because more complex models such as a model with co-activators and co-repressors would have an $O(n^5)$ complexity whose implementation time would be on the scale of years instead of hours [33]. Therefore, Ressoma et al. [35, 34] introduced temporal gene expression clustering as a data processing step into an improved gene interaction algorithm to reduce computation time. Such processing is based on a fact that if a gene combination of cluster centers does not fill the fuzzy-logic model well, it is unlikely that any genes with similar expression profiles will fit the model well. Therefore, after clustering gene expression data by self-organizing maps (SOM), only cluster centers are needed to fit the model, which substantially reduce the gene combinations. Instead of

only using fuzzy 'AND' operations for rule aggregation, Ressoma et al. [35, 34, 33] tried a few fuzzy models with different conjunction and rule aggregation operations such as Mamdani's model, Kosko's Standard Additive Model (SAM), a hybrid fuzzy model that combined attributes of the Mamdani's model and SAM. The experimental results on several real gene expression datasets show that the improved fuzzy-logic model is more efficient for reconstructing gene regulatory networks in terms of computation time and the Mamdani's model performs best in terms of the resilience to noise. Another generalization of Woolf and Wang's method is given in [31] which predicts changes in expression level of the target over interval time points and allows a wider search space for inferring regulatory relationships.

To represent necessary biological details in gene expression, Sokhansanj and Fitch [40] developed a fuzzy-logic approach to model gene regulation by using five fuzzy sets 'very low', 'low', 'medium', 'high', and 'very high'. A typical fuzzy rule has n inputs and one output like

$$\textbf{IF } \mathtt{P}_1 \text{ AND } \mathtt{P}_2 \text{ AND } \cdots \text{ AND } \mathtt{P}_n, \textbf{THEN } \mathtt{Q}$$

However, such fuzzy rule has a problem of curse of dimensionality, i.e. if the inputs P_i have M fuzzy sets, this requires a rule base with M^n rules. Although clustering is useful in combining proteins that are co-regulated and reduce the possible combinations, it defeats the purpose of a reasonably detailed gene regulation model [40]. To solve this problem, in [40], Union Rule Configuration (URC) which avoids combinatorial explosion in the fuzzy rule base [6] was used, in which the above rule would be written as

$$(\textbf{IF } \mathtt{P}_1 \textbf{THEN } \mathtt{Q}) \text{ OR } (\textbf{IF } \mathtt{P}_1 \textbf{THEN } \mathtt{Q}) \cdots \text{ OR } (\textbf{IF } \mathtt{P}_n \textbf{THEN } \mathtt{Q})$$

With this form, only $M \cdot n$ rules are required, which makes rule evaluation computationally feasible and mining data to obtain the rule base quickly. URC is likely not equivalent to the original formulation in fuzzy logic, but it succeeds as a heuristic method in many problems. Sokhansanj et al. [41] introduced a scalable linear variant of fuzzy logic for gene regulatory network modeling and used an exhaustive search for fuzzy gene interaction models that best fit transcription microarray measurements. Datta and Sokhansanj [8] built on above linear fuzzy logic model and employed an evolutionary search algorithm instead of exhaustive search to accelerate the search for finding a biomolecular network model as consistent with biological data as possible. In [41, 8], the magnitude of gene expression was normalized to $[-1, 1]$ and represented by the fuzzy sets 'low', 'medium', 'high' with the following membership functions as fuzzification scheme:

$$\mu_{\mathtt{low}}(x) = \begin{cases} -x & x < 0, \\ 0 & x > 0, \end{cases} \tag{9.1}$$

$$\mu_{\mathtt{medium}}(x) = 1 - |x|, \tag{9.2}$$

$$\mu_{\texttt{high}}(x) = \begin{cases} 0 & x < 0, \\ x & x > 0. \end{cases} \tag{9.3}$$

To bridge quantitative and qualitative biological data, the simplified centroid method was adopted as defuzzification scheme:

$$\widetilde{x} = \frac{\mu_{\texttt{high}} - \mu_{\texttt{low}}}{\mu_{\texttt{low}} + \mu_{\texttt{medium}} + \mu_{\texttt{high}}}.$$

Then, fuzzy predictions generated by a set of fuzzy rules were compared with quantitative data and evaluated by

$$E = \frac{\sum_{j=1}^{m}(x_j - \widetilde{x}_j)}{\sum_{j=1}^{m}(x_j - \bar{x}_j)}$$

where m is the number of data points in the time series of gene expression profiles and $\bar{x}$ is the average expression ratio over the whole series. With this criteria, exhaustive search or evolutionary search algorithm was adopted to find the best-fit fuzzy model (a set of fuzzy rules) which represents the gene regulatory network most consistent with biological data. In a similar framework, Linden and Bhaya [23] designed an evolutionary algorithm to find a set of fuzzy rules that could represent the actual regulation of gene expression. Unlike other "black-box" software, fuzzy-logic based gene network inference approaches can be understood and applied by biologists. A major advantage of fuzzy logic model is that it can tolerate the noise underlying gene expression data by qualitative representation of expression levels.

In addition to fuzzy logic, fuzzy adaptive resonance theory (fuzzy ART) also was used as a modeling method for gene regulatory networks. Fuzzy ART is a type of unsupervised clustering method and introduced by Carpenter et al. [5]. Takahashi et al. [42] developed a fuzzy ART associated matrix method for inferring gene regulatory networks. In this method, fuzzy ART was used to cluster gene expression data, which can decrease the number of time course gene expression patterns and reduce the implementation time. Fuzzy ART matrix is based on the assumption that a gene with an early maximum gradient point in the expression pattern will influence or regulate the gene in the same cluster or the gene with a proximate maximum gradient point. So, in [42], gene clusters were arranged according to the order of maximum gradient points of expression patterns, based on which regulatory relationships can be figured out (Fig. 9.5). In addition, 2D matrix was constructed to extract some common gene interactions under different stress conditions. Note that many feedback loops may be ruled out by this method since it exacts gene-gene interactions according to the order of clusters.

Finally, like their applications in analyzing gene expression data, fuzzy clustering algorithms have been used to reconstruct gene regulatory networks. The basic idea underlying this class of methods is that some genes interact between multiple regulatory pathways and soft clustering algorithm can account for this. Sehgal et al. [38] presented an framework called Collateral-Fuzzy Gene Regulatory Network Reconstruction (CF-GeNe) for gene regulatory network (GRN)

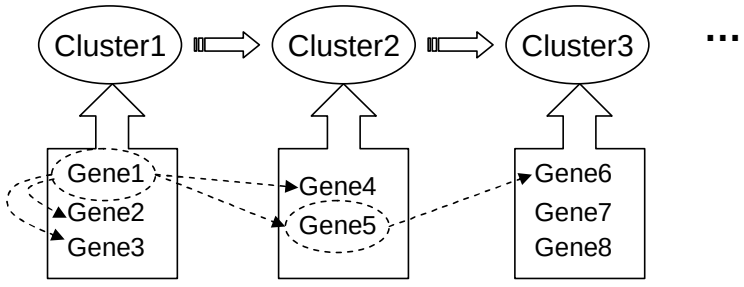


Fig. 9.5. Extracting gene regulatory interactions from the order of clusters

inference. Specifically, the fuzzy-PBM index is used to automatically determine the cluster number. Then fuzzy c-means was employed to find expression patterns (gene clusters) based on which significant genes with large inter-class variations and small intra-class variations were selected for gene network inference by using the Between Group to within Group Sum of Squares (BSS/WSS) method. The last step is that Spearman rank correlation is computed between each gene G_i in the selected significant genes set and all genes G_j within the cluster by

$$\rho = \frac{6 \sum D_g^2}{N_g(N_g^2 - 1)},$$

where D_g is the distance between the ordered gene pair G_I and G_J, while N_g is the number of pairs. The resulting gene pairs with high correlation compose of gene regulatory networks. By a same research group, an adaptive fuzzy evolutionary GRN Reconstruction (AFEGRN) framework with a similar spirit for modeling GRNs was presented [39]. Du et al. [12] proposed a new multi-scale fuzzy c-means method to cluster gene expression data and find co-regulated genes. Then, the time correlation between coregulated genes A and B can be expressed in discrete form as

$$R_{AB}(\tau) = \sum_{t=1}^{n} x_A(t) x_B(t - \tau),$$

where x_A and x_B are normalized (zero mean, standard deviation of unity) expression profiles of genes A and B, τ is the time shift. For multiple data sets, say L data sets, the time correlation results of each data set are combined as

$$R_{AB}^C(\tau) = \sum_{k=1}^{L} R_{AB}^k(\tau).$$

By using $\max_\tau R_{AB}^C(\tau)$, the time delay τ' between expression profiles of genes A and B can be estimated and possible regulatory relationships can be obtained according to the following rules:

- $R^C_{AB}(\tau) > 0, \tau' \neq 0$, there is positive regulation between genes A and B;
- $R^C_{AB}(\tau) < 0, \tau' \neq 0$, there is negative regulation between genes A and B;
- $R^C_{AB}(\tau) > 0, \tau' = 0$, genes A and B are positively co-regulated;
- $R^C_{AB}(\tau) < 0, \tau' = 0$, genes A and B are negatively co-regulated.

where the sign of τ' determines the direction of regulation. $\tau' > 0$ means gene B regulates gene A with time delay τ'; $\tau' < 0$ means gene A regulates gene B with time delay. Maraziotis et al. [25] developed a recurrent neuro-fuzzy method to extract fuzzy rules from microarray data. The neuro-fuzzy method combine the advantages of the strong computational power and low-level learning of neural networks and the high-level human-like reasoning of fuzzy systems. In addition, the dynamic aspects of gene regulatory interactions are expressed by the recurrent structure of the neuro-fuzzy model. Other work on using fuzzy logic methods to infer gene regulatory networks can be found in [24].

9.4 Fuzzy Clustering Techniques for Gene Expression Data

The adaptability of cells and the diversity in cellular processes are accomplished through the cooperation and multi-functionality of groups of genes/proteins. Depending on the cellular environment, groups of genes are often co-expressed and each group is regulated by a specific mechanism. DNA microarray technology has the potential to create huge datasets in short times which require computational methods for analyzing such data. Clustering based on the assumption that a population of objects can be subdivided into smaller subgroups has proved to be an important tool for the purpose of identifying groups of genes or samples displaying a similar expression profile. Such partitioning has the main scope of facilitating data visualization and interpretation, and can be exploited to gain insight into gene regulatory networks underlying a biological process of interest and uncover potential biological mechanisms.

A number of methods have been developed to deal with the complex relationships between objects and have been applied to microarray data analysis [16, 10]. However, due to the complex nature of biological systems and the noises underlying data, gene expression datasets tend to have very diverse structures. Some of them even do not have well-defined clustering structures. Classical clustering techniques such as k-means, hierarchical clustering and self-organizing maps (SOM), typically construct clusters on the basis of pairwise distance between genes. As a consequence, they may fail to reveal nonlinear relationships between gene expression profiles, and thereby fail to correctly represent a dataset with nonlinear structure. More sophisticated clustering approaches have been developed specifically for microarray data clustering, such as GeneClust [11] and CLIFF [47]. Though in some particular cases they perform better than classical methods, none of them is proved consistently better across multiple different datasets [17]. Moreover, high algorithmic complexity severely limits their use and the traditional algorithms remain more popular for their conceptual simplicity

and easy implementation. Particularly, since the work of Eisen *et al.* [13], hierarchical clustering remains the most widely used clustering algorithm, although it has been described to suffer from a number of limitations mostly deriving from the local decision making scheme that joins the two closest genes or clusters without considering the data as a whole [43].

While these classical algorithms can accurately identify distinct expression patterns by grouping genes with similar expression behavior, they are unable to identify genes whose expression levels are similar to multiple, distinct groups of genes, ignoring the information about the inter-relatedness of genes. In other words, when analyzing large gene-expression datasets collected under various conditions, where genes are likely to be co-expressed with multiple groups of genes under different conditions, hard clustering methods cannot recognize such gene and result in inaccurate clusters which lead to incorrect conclusions about gene product behavior [14]. In addition, conventional partition clustering methods force all genes into clusters, even those for which the variations in expression do not fit into any global pattern. These methods will assign to each cluster some genes which may only be marginally relevant for the biological significance of the cluster. For such cases, fuzzy clustering techniques have been taken into consideration because of their capability to assign one gene to multiple clusters (fuzzy assignment), which allow to capture genes involved in multiple transcriptional programs and biological processes and can reveal additional information concerning gene co-expression such as overlapping clusters and overlapping cellular pathways [16, 10]. In other words, fuzzy logic methods assign a relative likelihood for each gene belonging to each cluster, which can characterize the multi-functionality of a gene. Such likelihood can be used to select genes exhibiting tight association to given clusters and allows us to focus only on genes which show coherent behavior within clusters. In addition, fuzzy logic methods inherently account for noise in the data because they use qualitative representation of gene expression rather than the precise values.

The fuzzy clustering method connects each gene to all clusters by an indicator vector, whose elements correspond to the membership degrees of the gene to all clusters, where the membership degree takes a value between 0 and 1. A membership degree close to 1 indicates that the gene has a strong association to the cluster, whereas a membership close to 0 indicates a weak association. The general goal of the fuzzy clustering method is to find a fuzzy partition matrix W, and assign each gene to some clusters according to W such that each gene belong to one or more clusters with different membership degrees.

The first choice for fuzzy clustering in real applications is the fuzzy c-means algorithm (FCM) which is the fuzzy logic extension of the hard clustering technique k-means. FCM searches for the membership degrees and centroids until there is no further improvement in the objective function value and it computes fuzzy assignment essentially on the relative distance between one object and all cluster centroids [4, 16, 10, 3]. Many variants of FCM have been proposed in the past years, including a heuristic variant that incorporates principle component analysis (PCA) and hierarchical clustering [16], and fuzzy j-means that

applies variable neighborhood searching to avoid cluster solution being trapped in local minima [3]. A FuzzySOM approach was also developed to improve FCM by arraying the cluster centroids into a regular grid [29]. Fuzzy j-means (FJM) developed recently by Belacel et al. [3] was inspired by the local search heuristic j-means which is developed for solving the minimum sum-of-squares clustering problem. J-means has better performance than the standard k-means, especially for clustering large datasets. In j-means and FJM methods, centroid moves belong to the neighborhood of the current solution defined by all possible centroid-to-pattern relocations. In FJM, the 'integer' solution is moved to the continuous one by finding centroids and membership degrees for all patterns and clusters [3].

The partition matrix $W = (w_{ik})$ in fuzzy clustering techniques is of size $n \times c$, where w_{ik} is the membership value of gene i $(i = 1, ..., n)$ for the cluster k $(k = 1, ..., c)$. The gene expression data can be represented by an $n \times m$ matrix:

$$\begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{pmatrix}$$

where n is the number of genes and m is the number of arrays. The x_{ij} represents the normalized, expression level of gene i in array j. In the following, we will describe FCM and its variants in a detailed manner.

Fuzzy c-means clustering: Fuzzy c-means clustering (FCM) is a fuzzy logic extension of classic k-means method which can be represented as follows [4]:

$$\min \quad J(W, V) = \sum_{i=1}^{n} \sum_{k=1}^{c} w_{ik}^{m} \|x_i - v_k\|^2,$$

where $J(W, V)$ represents the objective function reflecting the quality of the clustering obtained from prototypes V and membership W, and m is the degree of fuzzification (the choices of m is critical). The membership degrees w_{ik} are defined such that $0 \leq w_{ik} \leq 1$, under the constraint of $\sum_{k=1}^{c} w_{ik} = 1$ for $i = 1, \cdots, n$. $V = (v_1, v_2, \cdots, v_c)$ is the vector of cluster centers or prototypes, and $\|x_i - v_k\|^2$ is the Euclidean distance between gene i and the prototype of cluster k. The FCM approach is an unsupervized approach, always converges and tends to assign low membership degrees for noisy points.

Fuzzy j-means clustering: The FJM method, introduced by Belacel et al. [3], uses all possible centroid-to-pattern relocations in order to construct move-defined neighborhoods. Fuzzy j-means clustering (FJM) method is a fuzzy logic extension of j-means. As the fuzziness parameter $m = 1$ defines a hard clustering, the m parameter for fuzzy logic applications has to be $m > 1$. The objective function of FCM can therefore be reformulated to:

$$\min \quad R(V) = \sum_{i=1}^{n} \left[\sum_{k=1}^{c} \|x_i - v_k\|^{2(1-m)} \right]^{1-m},$$

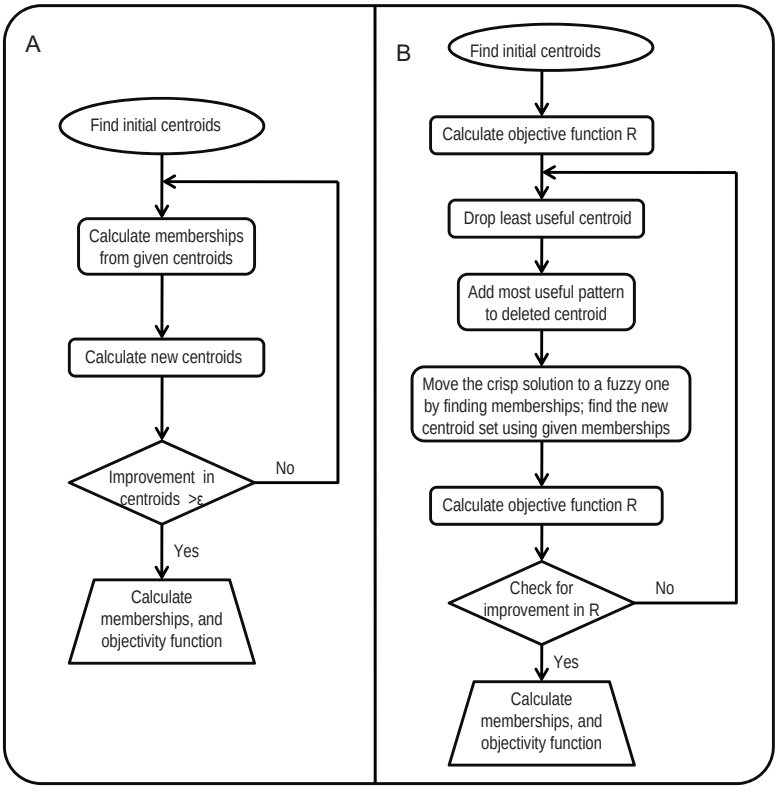


Fig. 9.6. Comparison of algorithm flowchart for FCM and FJM (adapted from [3])

where $R(V)$ is the reformulated objective function which is only related to the centroid positions. Therefore, centroid positions can be obtained directly by minimizing the above modified objective function. The obtained centroids can be used to calculate the membership values, and the results can subsequently be iteratively improved.

The algorithms describing FCM and FJM are briefly outlined in Fig. 6. Membership values and centroids of FJM are calculated in the same way as in FCM [3]. Both FCM and FJM are local heuristic algorithms and can therefore only determine a solution closest to the starting center which may not be an optimal solution. As a matter of fact, all those fuzzy c-means-derived clustering approaches suffer from the same basic limitation underlying k-means, i.e. using pairwise similarity between objects and cluster centroids for membership assignment, thus lacking the ability to capture non-linear relationships.

Determination of the fuzziness parameter m: Note that the criterion $J(W, V)$ depends on the choice of fuzziness parameter m. According to the analysis of Dembele and Kastner [10], it is not appropriate for fuzzification parameter to be set to a common value of 2 when the fuzzy clustering method is applied

to microarray data analysis. Thus, an initial step in any further investigation of fuzzy clustering methods is how to determine an optimal value of m for the studied dataset [10, 3]. Many applications of the FCM method have been hindered so far by problems associated with the choice of m. Dembele and Kastner [10] proposed a novel method which computes an upper bound value for m, and is then used to choose m independently of the number k of clusters before applying the clustering algorithm. Its related computation uses only first and second-order sample statistics (mean and variance) of distances between genes. Belacel *et al.* [3] also suggested two empirical rules for the estimation of optimal m. Previous studies used a hypothesis test which allows to reveal clustering structure in a given data set. In contrast, the computation for an upper bound value for m does not require knowledge of the distribution of distances as in the case of the hypothesis test. But we should note that the optimal m values given in such way are different for different datasets [10, 3, 14].

Another class of fuzzy clustering approaches is based on Gaussian Mixture Model (GMM) combined with expectation-maximization schemes [30, 48], where the dataset is assumed to be generated by a mixture of Gaussian distributions with certain probability, and an objective function is calculated based on the mixture Gaussians as the likelihood of the dataset being generated by such model. Then the objective function is maximized to solve the model and give a set of probabilistic assignments. A possible problem in this approach, as highlighted by Yeung and colleagues [48], is that real expression data do not always satisfy the basic Gaussian Mixture assumption even after various transformations aimed at improving the normality of the data distributions.

Recently, Fu and Medico [14] proposed a novel and powerful clustering algorithm named fuzzy clustering by local approximation of membership (FLAME). FLAME is mainly based on two general assumptions. One is that clusters should be identified in the relatively dense part of the dataset, the other is that neighboring objects with similar features (expression profiles) must have similar cluster memberships so that the membership of one object is constrained by the memberships of its neighbors. Therefore, the membership of each single object (gene or sample) is not determined with respect to all other objects in the dataset or to some cluster centroids, but determined with respect to its neighboring objects only. In contrast to general fuzzy clustering algorithms, this approach brings the notable advantage of capturing non-linear relationships between different genes or samples, in a way similar to a nonlinear data dimensionality reduction approach called locally linear embedding [36]. After this, the dataset can be represented in a lower dimensional space, where each object is mapped according to the lower dimensional representation of its nearest neighbors and the weights are assigned to its nearest neighbors. In this way the local structure of the original dataset (the neighbors of each object and their proximity) is preserved in a lower dimensional space. Then a fuzzy clustering approach based on neighborhood approximation can be adopted for capturing non-linear relationships in multidimensional data and providing a substantial improvement in the visualization and analysis of microarray data.

The authors have empirically estimated its computational efficiency by analyzing its time complexity and performing an empirical study of the time complexity of FLAME compared with other algorithms. The empirical result shows that for data matrices with many columns, FLAME has significant computational advantage over other methods, except k-means. But actually in traditional implementation, no sophisticated techniques are implemented for k-means to search for global minimum, while FLAME usually guarantees global minimum. Taking this into account, k-means may not have many computational advantages over FLAME.

The fuzzy clustering techniques offer several benefits for researchers. First, it generates accessible internal cluster structures, i.e. it indicates how well corresponding clusters represent genes. Second, the overall relation between clusters, and thus a global clustering structure, can be defined. The fuzzy clustering techniques can also be useful for unraveling complex modes of regulation for some genes. It is indeed known that some genes are subject to regulation by several molecular pathways. The overall expression patterns for such genes may therefore correspond to the superimposition of distinct patterns, each corresponding to a given mode of regulation. In this respect, the clusters defined by the second or third highest membership value can identify such secondary modes of regulation.

Another reason for applying fuzzy clustering is the high level of noise in microarray data due to numerous biological and experimental factors [15]. A common procedure to reduce noise in microarray data is to set a minimum threshold for expression changes. Genes below this threshold are excluded from further analysis. However, the determination of such exact threshold value seems to be arbitrary due to the lack of an established error model. Additionally, filtering may exclude interesting genes from further analysis. Soft clustering is a valuable approach here since it is highly robust to noise and pre-filtering can be avoided. Note that microarray data normalization is basically an important step for obtaining data that are reliable and usable for subsequent analysis [20]. Studies have shown that the choice of different normalization methods drastically affects the result of the cluster analysis [49]. In summary, we would emphasize that these two factors that affect clustering results should be paid attention to when fuzzy clustering methods are applied to the microarray data analysis successfully.

9.5 Fuzzy Systems in Analyzing Protein Interaction Networks

Biological networks have long been considered organized in a modular manner, which is composed of topologically or/and functionally isolated subnetworks corresponding to specific biological units [51]. Generally, modules can be understood as subnets which are densely connected within themselves but sparsely connected with the rest of the network. Revealing modular structure in cellular networks is helpful for understanding biochemical processes and signal pathways. Note that

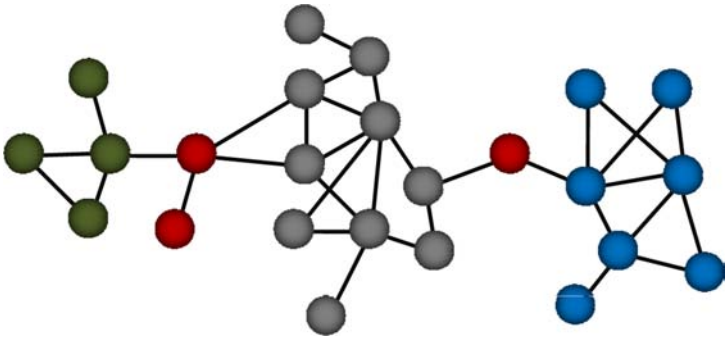


Fig. 9.7. A toy network with three nodes lies between modules

module structure referred to as "community structure" in the field of complex system theory also has attracted great interests [27, 7]. Many community-detection algorithms have been developed before in the field of complex networks. However, most current methods are hard partition algorithms which mean that each protein belongs to only one specific module. Such algorithms are not suitable for finding overlapping modules (See Fig. 9.7). Although some local search methods can detect modules with overlap, there is no detailed discussion on the possible significance and biological implication of overlapping nodes. Recently, some methods have been developed particularly for detecting fuzzy modules [32, 28, 53, 54, 26]. For example, the so-called CFinder method [28] has been applied for detecting overlapping functional modules in protein interaction networks [1, 19]. But the CFinder method is too restrictive and its basic element is 3-clique, and thereby it can detect few modules with many nodes excluded, especially in sparse protein interaction networks [52]. The Potts model for fuzzy community detection [32] is a random search procedure and returns different assignments of nodes upon different initial assignments. They repeat the algorithm many times and combine the inconsistence of the resulted assignments to form fuzzy communities.

Like applications in other fields, several fuzzy logic-based methods have been developed for detecting community structure in complex networks or functional modules in protein interaction networks [53, 54, 26]. Similar with the partition matrix W in the FCM method, a partition matrix $U = [u_{ik}]_{N\times c}$ which has the same constraints with W is needed. Partitions of this type on networks are called fuzzy partitions. The fuzzy membership degrees for a given vertex can be thought as a feature vector that describes some properties of the entity in a compact manner. However, these methods usually require a distance function defined in the data space, therefore it is impossible to apply them to graph partitioning directly, except in cases where the vertices of the graph are embedded in an n-dimensional space.

Recently, Zhang *et al.* [53] discusses a possible embedding of the vertices of an arbitrary graph into an n-dimensional space using spectral mapping in order to utilize the fuzzy c-means (FCM) algorithm on graphs. Specifically, a generalized

modular function $\tilde{Q}$ employing fuzzy concept is introduced, which is devised for evaluating the goodness of overlapping community structure. They also map data points (i.e. network nodes) into Euclidean space by computing the top K eigenvectors of a generalized eigen system.

The flow of the algorithm: Given an upper bound K of the number of clusters and the adjacent matrix $A = (a_{ij})_{n\times n}$ of a network.

The detailed algorithm is stated straightforward for a given λ as follows:

- Spectral mapping:
  1. Compute the diagonal matrix $D = (d_{ii})$, where $d_{ii} = \sum_k a_{ik}$.
  2. Form the eigenvector matrix $E_K = [e_1, e_2, \cdots, e_K]$ by computing the top K eigenvectors of the generalized eigen system $Ax = \lambda Dx$.
- Fuzzy c-means: for each value of k, $2 \leq k \leq K$
  1. Form the matrix $E_k = [e_2, e_3, \cdots, e_k]$ from the matrix E_K.
  2. Normalize the rows of E_k to unit length using Euclidean distance norm.
  3. Cluster the row vectors of E_k using fuzzy c-means or any other fuzzy clustering method to obtain a soft assignment matrix U_k.
- Maximizing modular function: Pick the k and the corresponding fuzzy partition U_k that maximizes $\widetilde{Q}(U_k)$.

In the algorithm above, the FCM method is initialized such that the starting centroids are chosen to be as orthogonal as possible which does not increase the time complexity, and can reduce the need for restarting the random initialization process and improve the quality of the clusterings. It can identify meaningful fuzzy communities in several well-known networks, but the eigenvector calculation involved in the algorithm renders it computationally expensive to use on large networks. Note that this method is also sensitive to the exact value of fuzziness parameter m. It can be determined by the empirical estimation [10, 3].

To overcome the need of spatial embedding, Nepusz *et al.* [26] proposed a different approach based on vertex similarities. A meaningful partition should group vertices that are somehow similar to each other in the same community. It is reasonable to assume that an edge between vertex v_1 and v_2 implies the similarity of v_1 and v_2, and likewise, the absence of an edge implies dissimilarity. Suppose we have a prior assumption about the actual similarity of the vertices, denoted by $\tilde{s}_{ij}$ for v_i and v_j. Let s_{ij} denote the similarity function based on the partition matrix U. The following equation measures the fitness of a given partition U of graph $G(V, E)$ by quantifying how precisely it approximates the prescribed similarity values with s_{ij}:

$$D_G(U) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\tilde{s}_{ij} - s_{ij})^2,$$

where w_{ij} are optional weights and n is the number of vertices in the network. Let $S(U) = [s_{ij}]$ and $\tilde{S} = [\tilde{s}_{ij}]$ and assume that $\tilde{S} = A$, the adjacency matrix of the network, such that the similarity of connected vertex pairs should be close to 1 and the similarity of disconnected vertex pairs should be close to zero. The

authors define a similarity function s_{ij} that satisfies the conditions prescribed above as follows:

$$s_{ij} = \sum_{k=1}^{c} u_{ki} u_{kj}.$$

It easily follows that $S(U) = [s_{ij}] = U^T U$. In summary, the community detection problem in this framework boils down to the optimization of $D_G(U)$ defined above. The goal is to find U that minimizes $D_G(U)$ while satisfying the general constraints of partition matrix. The number of clusters c, the weight matrix W and the desired similarities $\tilde{S}$ are given in advance (the adjacency matrix A is most commonly chosen as the latter matrix). This is a nonlinear constrained optimization problem which suggested to be solved by a gradient-based iterative optimization method.

One of the advantages of fuzzy community detection is that it enables us to analyze to what extent a given vertex is shared among different communities. A measure called bridgeness of a vertex v_i was defined as the distance of its membership vector $u_i = [u_{1i}, u_{2i}, \ldots, u_{ci}]$ from the reference vector $[\frac{1}{c}, \frac{1}{c}, \ldots, \frac{1}{c}]$ in the Euclidean vector norm, inverted and normalized to the interval $[0, 1]$ as follows:

$$b_i = 1 - \sqrt{\frac{c}{c-1} \sum_{j=1}^{c} \left(u_{ji} - \frac{1}{c} \right)^2}.$$

A vertex that belongs to only one of the communities has zero bridgeness, while a vertex that belongs to all of the communities exactly to the same extent has a bridgeness of 1.

Recently, Zhang *et al.* [54] presented an interesting community detection method based on non-negative matrix factorization (NMF) technique. Based on a quantitative function, a proper feature matrix from diffusion kernel and NMF algorithm, the presented method can detect an appropriate number of fuzzy communities in which a node may belong to more than one community. The distinguished characteristic of the method is its capability of quantifying how much a node belongs to a community. The FCM method can only give a relative membership which means that the membership of a node to a cluster is related to its membership to other clusters since the sum of its membership to all clusters equals to 1, whereas the NMF method quantifies how much a node belongs to a community in an absolute membership manner which is more reasonable since it can reflect the absolute possibility that a node belongs to a specific community. It has been applied to a protein interaction network and the preliminary results have shown its usefulness.

9.6 Summary

Uncertainty is considered essential to science. Fuzzy logic, as a qualitative computational approach, is a way to model and deal with data using natural language. Since uncertainty is inherent in biological data due to uncertainty of

biological experiments and other factors, fuzzy logic and fuzzy set based theory and techniques can be considered as a suitable formalism to deal with the imprecision intrinsic to many biological problems. In this chapter, we briefly introduce the basic principle of gene expression, gene expression databases as well as the concepts of gene regulatory networks and protein interaction networks. Then we comprehensively summarize the application of fuzzy logic and fuzzy logic-based clustering techniques in modeling gene regulatory networks and analyzing gene expression data and protein networks. Beyond all doubt, fuzzy logic based methods have shown powerful effectiveness in many aspects. But even though, we think that this is only a start. The fuzzy logic model and fuzzy clustering techniques are expected to take a key role in analyzing and modeling the huge amount of microarray datasets and grouping proteins/genes based on network topological structure in the future.

Acknowledgements

This work is partly supported by the National Basic Research Program (973 Program) under Grant No.2006CB503900, National Natural Science Foundation of China under Grant No.1063107, No.10701080 and Joint Research Grant supported by NSFC and JSPS under Grant No.10711140116.

References

1. Adamcsek, B., Palla, G., Farkas, I.J., Derényi, I., Vicsek, T.: CFinder: Locating cliques and overlapping modules in biological networks. Bioinformatics 22, 1021–1023 (2006)
2. Akutsu, T., Miyano, S., Kuhara, S.: Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In: Pacific Symposium on Biocomputing, vol. 4, pp. 17–28 (1999)
3. Belacel, N., Cuperlovic-Culf, M., Laflamme, M., Ouellette, R.: Fuzzy j-means and vns methods for clustering genes from microarray data. Bioinformatics 20, 1690–1701 (2004)
4. Bezdak, J.C.: Pattern recognition with fuzzy objective function algorithms. Plenum Press, New York (1981)
5. Carpenter, G.A., Grossberg, S., Rosen, D.B.: Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. Neural Networks 4, 759–771 (1991)
6. Combs, W.E., Andrews, J.E.: Combinatorial rule explosion eliminated by a fuzzy rule configuration. IEEE Trans. Fuzzy Syst. 6, 1–11 (1998)
7. Danon, L., Duch, J., Diaz-Guilera, A., Arenas, A.: Comparing community structure identification. J. Stat. Mech., P09008 (2005)
8. Datta, S., Sokhansanj, B.A.: Accelerated search for biomolecular network models to interpret high-throughput experimental data. Bioinformatics 8, 258 (2007)
9. de Jong, H.: Modeling and simulation of genetic regulatory systems: a literature review. J. Comput. Biol. 9, 67–103 (2002)
10. Dembele, D., Kastner, P.: Fuzzy c-means method for clustering microarray data. Bioinformatics 19(8), 973–980 (2003)

11. Di Gesu, V., Giancarlo, R., Lo Bosco, G., Raimondi, A., Scaturro, D.: GenClust: a genetic algorithm for clustering gene expression data. BMC Bioinformatics 6, 289 (2005)
12. Du, P., Gong, J., Wurtele, E.S., Dickerson, J.A.: Modeling gene expression networks using fuzzy logic. IEEE Transactions on Systems, Man, and Cybernetics, Part B 35(6), 1351–1359 (2005)
13. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. USA 95, 14863–14868 (1998)
14. Fu, L., Medico, E.: FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data. BMC Bioinformatics 8(3) (2007)
15. Futschik, M.E., Charlisle, B.: Noise robust clustering of gene expression time-course data. Journal of Bioinformatics and Computational Biology 3(4), 965–988 (2005)
16. Gasch, A.P., Eisen, M.B.: Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. Genome. Biol. 3(11), 1–22 (2002)
17. Handl, J., Knowles, J., Kell, D.B.: Computational cluster validation in post-genomic data analysis. Bioinformatics 21(15), 3201–3212 (2005)
18. Husmeier, D.: Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic bayesian networks. Bioinformatics 19, 2271–2282 (2003)
19. Jonsson, P.F., Bates, P.A.: Global topological features of cancer proteins in the human interactome. Bioinformatics 22(18), 2291–2297 (2006)
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. Klir, G.J., Yuan, B. (eds.): Fuzzy sets, fuzzy logics and fuzzy systems — selected papers by lotfi a zadeh. World Scientific, Singapore (1996)
22. Kosko, B.: Global stability of generalized additive fuzzy systems. Transactions on Systems, Man, and Cybernetics 28(3), 441–452 (1998)
23. Linden, R., Bhaya, A.: Evolving fuzzy rules to model gene expression. BioSystems 88, 76–91 (2007)
24. Ma, P.C.H., Chan, K.C.C.: Inference of gene regulatory networks from microarray data: a fuzzy logic approach. In: Proceedings of Asia-Pacific Bioinformatics Conference, pp. 17–26 (2006)
25. Maraziotis, I., Dragomir, A., Bezerianos, A.: Recurrent neuro-fuzzy network models for reverse engineering gene regulatory interactions. In: Berthold, M.R., Glen, R.C., Diederichs, K., Kohlbacher, O., Fischer, I. (eds.) CompLife 2005. LNCS (LNBI), vol. 3695, pp. 24–34. Springer, Heidelberg (2005)
26. Nepusz, T., Petróczi, A., Négyessy, L., Bazsó, F.: Fuzzy communities and the concept of bridgeness in complex networks. Physical Review E 77, 016107 (2008)
27. Newman, M.E.J.: Detecting community structure in networks. Eur. Phys. J. B. 38, 321–330 (2004)
28. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature 435, 814–818 (2005)
29. Pascual-Marqui, R.D., Pascual-Montano, A.D., Kochi, K., Carazo, J.M.: Smoothly distributed fuzzy c-means: a new self-organizing map. Pattern Recognition 34, 2395–2402 (2001)
30. Qu, Y., Xu, S.: Supervised cluster analysis for microarray data based on multivariate Gaussian mixture. Bioinformatics 20, 1905–1913 (2004)
31. Ram, R., Chetty, M., Dix, T.I.: Fuzzy model for gene regulatory network. In: IEEE Congress on Evolutionary Computation, pp. 1450–1455 (2006)

32. Reichardt, J., Bornholdt, S.: Detecting fuzzy community structures in complex networks with a Potts model. Phys. Rev. Lett. 93, 218701 (1993)
33. Ressom, H., Natarajanb, P., Varghesea, R.S., Musavib, M.T.: Applications of fuzzy logic in genomics. Fuzzy Sets and Systems 152, 125–138 (2005)
34. Ressom, H., Reynolds, R., Varghese, R.S.: Increasing the efficiency of fuzzy logic-based gene expression data analysis. Physiol. Genomics 13, 107–117 (2003)
35. Ressom, H., Wang, D., Varghese, R.S., Reynolds, R.: Fuzzy logic-based gene regulatory network. In: IEEE International Conference on Fuzzy Systems, vol. 2, pp. 1210–1215 (2003)
36. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290, 2323–2326 (2000)
37. Schena, M., Shalon, D., Davis, R.W., Brown, P.O.: Quantitative monitoring of gene expression patterns with a complementary DNA microarray. Science 270, 467–470 (1995)
38. Sehgal, M.S.B., Gondal, I., Dooley, L.: CF-GeNe: Fuzzy framework for robust gene regulatory network inference. Journal of Computers 1(7), 1–8 (2006)
39. Sehgal, M.S.B., Gondal, I., Dooley, L., Coppel, R.: AFEGRN: Adaptive fuzzy evolutionary gene regulatory network re-construction framework. In: IEEE International Conference on Fuzzy Systems, pp. 1737–1741 (2006)
40. Sokhansanj, B.A., Fitch, J.P.: URC fuzzy modeling and simulation of gene regulation. In: Annual International Conference of the IEEE Engineering in Medicine and Biology Society, vol. 3, pp. 2918–2921 (2001)
41. Sokhansanj, B.A., Fitch, J.P., Quong, J.N., Quong, A.A.: Linear fuzzy gene network models obtained from microarray data by exhaustive search. BMC Bioinformatics 5, 108 (2004)
42. Takahashi, T., Tomita, S., Kobayashi, T., Honda, H.: Inference on common genetic network using fuzzy art associated matrix method. J. Biosci. Bioeng. 96, 154–160 (2003)
43. Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S., Golub, T.R.: Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. Proc. Natl. Acad. Sci. USA 96(96), 2907–2912 (1999)
44. Torres, A., Nieto, J.J.: Fuzzy logic in medicine and bioinformatics. Journal of Biomedicine and Biotechnology 2006, ARTICLE ID 91908 (2006)
45. Vinterbo, S.A., Kim, E.Y., Ohno-Machado, L.: Small, fuzzy and interpretable gene expression based classifiers. Bioinformatics 21, 1964–1970 (2005)
46. Woolf, P.J., Wang, Y.: A fuzzy logic approach to analyzing gene expression data. Physiol. Genomics 3, 9–15 (2000)
47. Xing, E.P., Karp, R.M.: CLIFF: clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. Bioinformatics 17(S1), S306–S315 (2001)
48. Yeung, K.Y., Fraley, C., Murua, A., Raftery, E., Ruzzo, W.L.: Model-based clustering and data transformations for gene expression data. Bioinformatics 17(10), 977–987 (2001)
49. Yeung, K.Y., Haynor, D.R., Ruzzo, W.L.: Validating clustering for gene expression data. bioinformatics. bioinformatics 17(4), 309–318 (2001)
50. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)
51. Zhang, S., Jin, G., Zhang, X.S., Chen, L.: Discovering functions and revealing mechanisms at molecular level from biological networks. Proteomics 7(16), 2856–2869 (2007)

52. Zhang, S., Liu, H.W., Ning, X.M., Zhang, X.S.: A hybrid graph-theoretic method for mining overlapping functional modules in large sparse protein interaction networks. International Journal of Data Mining and Bioinformatics (in press)
53. Zhang, S., Wang, R.S., Zhang, X.S.: Identification of overlapping community structure in complex networks using fuzzy c-means clustering. Physica A 374(1), 483–490 (2007)
54. Zhang, S., Wang, R.S., Zhang, X.S.: Uncovering fuzzy community structure in complex networks. Physical Review E 76, 046103 (2007)

10

Evolving a Fuzzy Rulebase to Model Gene Expression

Ricardo Linden[1] and Amit Bhaya[2]

[1] Faculdade Salesiana Maria Auxiliadora, CEPEL
rlinden@pobox.com
[2] COPPE-UFRJ
amit@nacad.ufrj.br

Summary. This chapter describes the use of genetic programming to evolve a fuzzy rule base to model gene expression. We describe the problem of genetic regulation in details and offer some reasons as to why many computational methods have difficulties in modeling it. We describe how a fuzzy rule base can be applied to this problem as well as how genetic programming can be used to evolve a fuzzy rule base to extract explanatory rules from microarray data obtained in the real experiments, which give us data sets that have thousands of features, but only a limited number of measurements in time. The algorithm allows for the insertion of prior knowledge, making it possible to find sets of rules that include the relationships between genes that are already known.

10.1 Biological Introduction

Biology has expanded its knowledge of the gene regulation process and a full description of known features of the regulation process would require a huge compendium. In this section we will describe the main features of gene regulation in order for the reader to understand the specific class of problems studied in this chapter. We will also describe the main characteristics of the data sets used in this problem.

10.1.1 Gene Regulation

All information necessary for the creation of the proteins that are necessary for the cell is coded in DNA. This information is extracted in a multi-step path that is called the central dogma of molecular biology that could be summarized as DNA $\longrightarrow$ RNA $\longrightarrow$ Protein, meaning that cell response and differentiation steps can be controlled in many different steps. This ability to respond to different needs and the basic process of differentiation are fundamental to an organism. No one would confuse a neuron with a liver or a heart cell, either based on their shape or on their function. Nevertheless, this difference is not due to major differences in their DNA sequence, but rather to differences in the expression level of each gene. In every cell, at every time point, only a fraction of the total DNA is expressed, that is, transcribed into mRNA, which is subsequently translated

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 191–215.
springerlink.com

into protein. The amount of proteins and other elements produced is regulated so that the necessary amounts are produced without depleting excessively the energetic reserves of the cell.

At every given time, a gene may be expressed at different levels, including zero (not expressed). Its expression level has a strong correlation to the amount of mRNA in the cell that is coded by the gene. It is important to understand that there are other control mechanisms in the cell, such as protein degradation speed, RNA transport and localization control *etc*, but for most cells the initiation of RNA transcription is the most important point of control [1]. Nevertheless, one should understand that considering only transcriptional control in the reverse engineering problem is a limitation of any method based on microarrays (see Section 10.1.2 for further details).

A cell can change the expression level of its genes in response to external signals. At any given time, the cell needs to inhibit or activate groups of genes in response to changing organism and environment requirements. In order to perform this regulation, cells have an elaborate mechanism that include several control areas in every gene, the catalysis of gene expression by several other substances and other mechanisms.

A gene can be regulated by several molecules and the ones it codes for can regulate one or more other genes, creating a pleiotropic regulation network, which occurs by the binding of molecules to gene control areas. Since many of these molecules are proteins synthesized based on the mRNA of other genes, we can say that one gene regulates another, even though DNA regions do not interact directly.

10.1.2 Microarray Data Sets

Nowadays, the data available on gene expression is growing exponentially. The creation of new tools has caused a genomics revolution, flooding scientists with huge amounts of data. DNA microarray technology is one of the technologies that has caused this huge impact in the biological sciences. It describes the state of each cell by measuring the mRNA expression levels, which, according to the central dogma, is a good approximation of protein levels. It is not precise however, because there are other mechanisms of control besides DNA transcription control, but measurement techniques for other substances such as proteins are not as accurate and widespread as microarrays.

A DNA array, or microarray, is defined as an orderly arrangement of tens to hundreds of thousands of unique radioactive or fluorescent DNA molecules (probes) of known sequence attached to a fixed surface. The DNA samples are then presented to the microarray and bind to their complementary sequences (hybridize), thus allowing for the determination of their sequence or their relative abundance (expression level) [26].

Microarrays allow one to study expression levels in parallel thus providing static information about gene expression (i.e., in which tissue(s) the gene is expressed) and dynamic information (i.e., how the expression pattern of one gene relates to those of others). The high degree of digital data extraction and

processing of these techniques supports a variety of samples or experimental conditions [6]. Microarray technology, as a high throughput approach of differential gene expression studies, efficiently generates massive amounts of gene regulation data, facilitating rapid identification of gene candidates to follow up with functional characterization.

There are, of course, some pitfalls when dealing with this technology that a cautious experimenter must deal with. We will not describe the problems in detail, for this is not the main goal of this chapter, but, in brief, the main problems with this technology are:

- Differences among samples
- Measurement errors from CCD or radioactivity detectors
- Normalization methods
- Background noise elimination
- Stochastic binding effects

A cautious experimenter will try to ensure that these problems are dealt with and that experiments are repeated in order to avoid the common errors associated with microarrays.

Once these problems are dealt with, the main problem with a microarray data set becomes its dimensions. Usually, thousands of genes are measured simultaneously for a small number of time steps. Given thousands of expression levels, it is a complex task to identify the differentially expressed genes and understand the myriad relationships that define the genetic regulation network. There are several well known statistical techniques for this [12], but they all require massive amounts of data. There are some techniques, such as those described in [34] and [25] that try to minimize the number of experiments, by reducing the dimensionality of the data, either with Fisher Discriminant Analysis (FDA), singular value decomposition (SVD) or by Wilks Lambda Score. Nevertheless, these techniques, even though they manage to reduce data dimensionality drastically, still require a significant number of measurements (of the order of tens, in both cases).

Unfortunately, given the high cost of experiments, this volume of data is not available. Thus, it is necessary to propose new techniques that can cope with lack of data and still mine the gene expression matrix for useful relationships. There is, of course, a price for dealing with such a small data set. First, we can point out to the loss of certainty, making it very difficult to make accurate statements about relationships between genes. Second, we incur the "blessing of dimensionality". Given the high number of genes measured and the small number of measurement points, we face a situation where there are many degrees of freedom and few constraints, so that there are many possible "good fit" solutions. Thus, it can be argued that any search method will stumble upon a reasonable solution, and the key is how to differentiate between various solutions. Therefore, any proposed method should be able to deal with both these problems.

10.2 Pros and Cons of Gene Regulatory Networks (GRN) Models

In this section we discuss some methods that are successful and popular in other fields of application but have inherent characteristics that make them less advisable in this specific application. This does not mean, of course, that these methods are of inferior quality or that it is impossible to obtain meaningful results with them, but only that their inherent characteristics make it more difficult to achieve good or reliable results when dealing with microarray data sets.

10.2.1 Boolean Networks

Perhaps the simplest model is the Boolean Network. Boolean networks were first introduced by Kauffman in the late 60's as an abstraction of genetic regulatory networks: each gene is modeled as being either "ON" or "OFF", and the state of each gene at the next time step is determined by a Boolean function of its inputs at the current time step [15].

Several papers use Boolean networks as a tool to model gene expression, either regular [21] or probabilistic [27]. This model assumes that each gene can take either 0 (not-expressed) or 1 (expressed) as its state value. This approach can model some aspects of real regulation, as discussed in [29]. Nevertheless, since gene regulation is gradual and varies with gene expression levels, the ability of Boolean Networks to model real genetic regulation is limited.

10.2.2 Neural Networks

The data set available in a typical gene regulatory network problem generally consists of hundreds to thousands of signals, usually measured for no more than twenty time-steps. This paucity of data renders network models inferred from this data statistically insignificant [31] and requires methods that can deal with this specific situation to produce results that have real value to "wet lab" experiment design, even though they may not be statistically significant for predictive purposes.

Artificial neural networks (ANN) are computational models that try to mimic, in an extremely simple way, the way the human brain works. Just like the brain, ANNs consist of a group of simple cells (artificial neurons) connected by synapses that form a massively parallel and distributed processing system. The structure of this system (connections and their weights) is formed during a training phase, when the ANN is presented with data examples, actually acquiring knowledge from its environment [18].

To perform this learning, neural networks rely on massive data sets to perform their training process, so they cannot be trained correctly, using traditional algorithms, in situations where data is scarce. Unfortunately, each experiment has a non negligible cost, so that it is financially not practical for most labs to perform a large number of experiments, which makes it impossible to generate the amount of data a neural network would require. When presented with small data sets, neural networks tend to simply memorize them, instead of actually learning the underlying structure that generated them.

10.2.3 Association Rules

Creighton and Hanash [7] describe a technique that uses association rules to mine for regulatory relationships among genes. The algorithm proposed uses data binning (actual discretization) to overcome measurement errors and inherent noise. The dangers associated with discretization are highlighted in [16] in which it is also proposed to search for association rules, using planes to separate data.

In [7] all expression levels are set to three different states: up, down and neither-up-nor-down, while in [5] all expression levels are transformed into a boolean variable (over-expressed and under-expressed). An idea that might improve these methods would be to adapt them to a fuzzy sets based approach, allowing them to have multiple fuzzy sets that would correspond to, for instance, highly expressed, very highly expressed, etc.

This technique suffers from being data intensive, because it needs several microarray runs to create frequent itemsets and establish a pattern of co-occurrences and, therefore, the cost limitations described above still apply. Using a small number of data points may cause the frequent itemsets to be relatively small, allowing for spurious effects (inherent to a multiply connected graph such as the biological regulation one) to create rules that do not have real importance.

This method, in common with all others that rely on large data sets, will improve as experimental costs begin to fall. As [16] points out, it is becoming increasingly common to see data sets with tens of experiments. In the future, when hundreds or even thousands of experiments become common, the ability of methods based on association rules to model data will improve considerably.

10.2.4 Linear Regression

If gene regulation could be assumed to be linear, then linear regression would be an effective tool to model it. However, there are commonly occurring biological effects that make the linearity assumption fail:

1. Saturation: at a concentration that is specific to each binding site and substrate, increasing the regulator concentration will not change the speed and/or the amount of regulatee action.
2. Catalysis: the presence of certain enzymes will change the speed of the reaction and this effect, together with saturation, will cause the reaction curve to be S-shaped. A piecewise linear method, such as described in [9], may minimize this as well as he effect mentioned in the previous item, by creating close linear approximations for each of the main function regions.
3. Inhibitory effects: The presence of certain substances will inhibit the regulation process, either shutting it down completely, or making it extremely inefficient. This effect can be modeled by combining a Boolean network with a linear one, as done in [4].

Because of all these effects, any technique that is based on a linearity assumption tends to yield poorer results. They can serve as a first approximation, but non-linearity will always insidiously decrease the veracity of their results, an effect that may be decreased if one combines a piecewise linear approach with Boolean networks.

10.2.5 Perturbation Analysis

The idea behind such methods, described in papers such as [30] and [3], is that additional information about a genetic network may be gleaned experimentally by applying a directed perturbation to the network, and observing the steady-state expression levels of every gene in the network in the presence of the perturbation.

The problem with this technique is similar to the one described for neural networks: perturbation experiments may become expensive, especially if they need to be repeated in order to eliminate spurious results. Another problem with this approach lies in the fact that there seems to be some evidence that nature rewards redundancy in the form of alternative pathways performing the same function, so that there are backups in case that one pathway fails. Therefore a knock-out experiment may not lead to the pathway being switched off, even if a seemingly crucial component of a pathway was knocked out, since a backup component is activated and takes over the function of the one knocked out.

Another issue is that in higher metazoa each gene is associated with an average of ten different biological functions [2]. This means that perturbation studies may disrupt several different pathways and cause unforeseen effects that may make it very difficult for the analyst to understand what is really happening in the single pathway he is studying.

10.2.6 Differential Equations

A genetic regulatory network can be understood as a dynamic system whose states are defined by the expression levels as measured by microarray experiments. As such, they could be described by differential or difference equation based models, which at once can be seen as superior to Boolean networks, given their ability to deal with intermediate expression levels.

The main problems with these models are the following:

- Discrete aspects: they cannot easily describe the discrete aspects of gene regulation such as binding of a transcription factor to the DNA, which is essentially an on/off event [4].
- Linear models vs Excessive data requirements: scientists tend to concentrate on linear models because they require less data, but at the cost of placing strong constraints on the nature of regulatory interactions in the cell that may lead to less accuracy in the regulatory interaction description [15]. The use of differential equations allows for the introduction of explicit rate constants and one could go beyond the linear additive summation and include higher order terms, the determination of the parameters of a network incorporating higher-order terms would require much more data than is generally available for these systems [32].
- If the algorithm uses time-series data, it must estimate the rates of change of the transcripts ($\frac{dx}{dt}$) from the series. This can be problematic because calculating the derivative can amplify the measurement errors (noise) in the data [15].

This section has not made an exhaustive list of methods applied to the gene regulation problem. Many other methods have been applied to this problem. The reader may refer to [35], for instance, to learn about other computational techniques to analyze genomic data.

10.3 Fuzzy Logic Applied to the Gene Regulation Problem

Now that we have discussed what methods should not be used, we can proceed to the discussion of the method proposed in this chapter. In this section we will describe the fuzzy logic model used to deal with microarray data.

10.3.1 Beneficial Characteristics of a Fuzzy Model

Fuzzy rules model naturally ill-posed problems characterized by very little information such as the one we are facing using a linguistic approach that is a form of information useful to human experts. Therefore, they can be used to convey imprecise information for experts that are available and can seed the systems with a number of effective rules from the outset and verify the linguistic results obtained [28]. Using this prior knowledge is a very interesting feature that we explored in this work and that becomes very difficult to implement in numeric methods, like differential equations and neural networks.

Whenever discovered knowledge is to be used to assist in decision-making by a human user, it is important that it be comprehensible to the user [13]. Given the discrepancy between data dimensions (high number of genes with small number of points) we cannot arrive at a definitive model, no matter what method is used. Therefore, the main purpose of purposing a fuzzy model is to create hypotheses that can subsequently be tested in the biological workbench ("wet lab"). This implies that simplicity of the rules is an essential feature.

Fuzzy rules also make it very easy to include delays in the model. It is enough to include a parameter t in the rule, where t stands for the delay. For instance, a rule with the format *IF HighlyExpressed(f_1 (-1)) AND LowlyExpressed(f_2 (-2)) THEN HighlyExpressed(f_3)* means that if in the previous time step the substance f_1 had a high expression level and two time steps before substance f_2 had a low expression level, then the substrate f_3 will have a calculated high expression value. The default value for t will always be 1, meaning that we are referring to the previous expression value available. Obviously, when we introduce this feature we create more degrees of freedom and the model must be able to deal with this issue.

10.3.2 Definition of the Fuzzy Sets

It is assumed that each gene may be associated with several fuzzy sets whose universes of discourse will cover all the numeric space that its expression levels can span.

In our work, the expression space of each feature is divided evenly, so that each fuzzy set has the same support. The process consists of the following steps:

1. Choose the number of fuzzy sets for the gene of interest (n_{fuzzy});
2. The universe of discourse for the variable is defined by reading the interval of expression values from the microarray data and expanding it a little (10%, for instance) in each direction;
3. The universe of discourse is then divided in ($n_{fuzzy} - 1$)parts;
4. Create the fuzzy sets using triangular functions, using the following scheme:
   - The first half part is the support of the first descending function;
   - The last half part is the support of the last ascending function;
   - The $n_{fuzzy}-2$ remaining parts are the supports of triangular membership functions.

We could assign meaning to each fuzzy set. For instance, if we divide the space into three fuzzy sets, they could be understood as representing *Low expression level*, *Medium expression level* and *High expression level*. If we divide the space in five fuzzy sets, we could interpret the fuzzy sets as meaning *Very low expression level*, *Low expression level*, *Medium expression level*, *High expression level* and *Very high expression level*.

There is overlapping among adjacent fuzzy sets, so that more than one rule may be active for each expression value.

10.3.3 Using Fuzzy Rules to Calculate Expression Levels

Having defined the fuzzy sets and the fuzzy rule base, calculating next time step expression levels is a simple matter of applying the current time step expression levels to the rules and defuzzifying the results. There are many defuzzification methods that could be used in this process and the choice of method is analogous to that used in any other fuzzy application.

In this application, we applied the medium of maxima (MoM) defuzzification method, expressed by the formula:

$$y_j = \frac{\sum_{i=1}^{n_{fuzzy}} \mu_i * y_i^{max}}{\sum_{i=1}^{n_{fuzzy}} \mu_i} \tag{10.1}$$

In this formula, n_{fuzzy} stands for the number of fuzzy sets defined for the gene of interest (see Section 10.3.2), μ_i is the calculated membership of the rule to the fuzzy set i and y_i^{max} is the expression value where the membership function of set i is at its maximum.

There may be rules that have zero support. In our experience, this may lead to poorer results, and a minimum degree should be designated for each rule before the defuzzification process. A small value, such as 0.05 will guarantee set participation and allow the algorithm to achieve better results.

10.3.4 Rule Introduction by an Expert

Fuzzy logic offers an appealing method for describing phenomena by a set of rules and data sets that are based on linguistic expressions very similar to the

ones we use daily to keep and transfer knowledge. These expressions include "high level of expression", "low level of expression" etc and the rules based on those concepts express knowledge in approximately the same way that a human expert would. An example of a fuzzy rule would be "if gene A is at a low level of expression, then gene B is at a high level of expression", which clearly means that gene A is an inhibitor to gene B.

Experts are already aware of many regulatory pathways and ignoring this prior knowledge is certainly wasteful, while exploiting it leads to a reduction in the size of the search space, which is always desirable.

It is not easy for an expert to incorporate prior knowledge into certain methods, such as differential equation based and neural networks based methods, which can be considered to be "numerically encoded", thus requiring translation of qualitative knowledge into appropriate quantitative knowledge, usually a difficult task. This is not the case with fuzzy systems. Since we have a rule base that can have many rules applied to a single fuzzy variable, experts can include their knowledge and this will be used together with the rules discovered by any algorithm. Using this prior knowledge makes the method smarter and also makes it possible for any tool created to become a hypothesis tester. In the next section we will discuss rule insertion by experts when describing a full genetic programming algorithm used to discover a fuzzy rule base.

10.4 Using Genetic Programming to Evolve a Fuzzy Rule Base

The basic concepts of using a fuzzy rule base were described in the previous section, but we still need an algorithm that can help us find what rules to use. In this section we will describe the evolutionary algorithm we use to create the fuzzy rule base that can best describe the gene regulation implicitly presented in the (small) data set.

10.4.1 Basic Concepts

Evolutionary algorithms (EA) are inspired by Nature. The idea is to mimic the natural evolution of the species using operators that simulate both sexual reproduction and random mutation in order to create a new kind of search technique that is robust and intelligently seeks solutions in a search space that may be too big for conventional techniques [22]. EA spawns several different techniques, including genetic algorithms (GA), genetic programming (GP) and evolutionary programming (EP).

Genetic Programming is a branch of evolutionary algorithms that simulates natural evolution to find complex structures such as programs and rules [19]. As with other EAs, it uses the concept of a population that reproduces and suffers mutation (as in natural, biological processes) and with high probability evolves toward a better solution that fits better to real data available. It is a heuristic that, although dependent on many probabilistic factors, tends to span much of

the solution space and is less affected by common data problems such as local minima in an energy function.

All evolutionary algorithms use a population of competing solutions subjected to random variation and selection for a specific purpose [10], which is to evolve the population to one that contains a higher proportion of superior (fitter) individuals. The fitness of each individual in the population (its quality) is a measure of how well that individual achieves the desired goal and is the main connection between the EA and the problem at hand. Therefore, the formulation must contain all aspects of the problem, including constraints.

The variation and selection are usually based on two operators, the crossover operator which combines two different individuals into a new one and the mutation operator, which randomly changes parts of one individual in order to increase diversity. Both are very important, representing two different aspects of the natural search: exploitation (using current solutions information to derive a new and possibly better solution), which is performed by the crossover operator, and exploration (venturing into new areas of the search space), which is performed by the mutation operator.

An evolutionary algorithm could be described by the following pseudo-code:

```
Create Initial Population
While termination criteria not met
    Select parents which will generate offspring from current population
    Apply genetic operators to the selected parents and generate offspring
    Select next population from current individuals and generated offspring
End While
Present best solution(s)
```

In this algorithm, termination is usually based on one or more of the following criteria:

- time based: a number of generations has elapsed;
- quality based: a certain performance has been achieved;
- stagnation based: the set of best individuals has not improved for a certain number of generations.

Parent selection must be done in a way so that best solutions have the bigger probability of reproducing, but not at the expense of preventing the worse solutions from also doing so. The idea is that these worse solutions may have important information coded in their "genes" that would be missed if they did not participate in the creation of new offspring. Therefore, a variation of a roulette approach is usually used in which the parents with highest evaluation ("fittest") correspond to a bigger fraction of the roulette wheel when a random "spin" of the wheel is performed.

Thus, in order to define an EA one must define the coding scheme (how each individual will be represented in the computer), the operators (both mutation and crossover and any other specific one that will be used), the evaluation or fitness function (i.e., a measure of the quality of the current solutions to the

problem at hand), the termination criteria used, the parent selection scheme and the next generation choice algorithm. We will discuss all these items applied to the problem of gene regulation reverse engineering in the following sections.

10.4.2 Dealing with Microarray Data Characteristics

The use of GP methods is justified by their ability to generate and test a broad spectrum of solutions, even from incomplete or insufficient data sets. As discussed in Section 10.1.2, microarray data usually consists of hundreds or thousands of genes whose expression levels are measured at only a few time points. This low dimensionality generally implies that statistical methods are liable to generate solutions of low statistical significance, but does not stop a GP method from generating testable hypotheses for the biology lab.

The high number of degrees of freedom of the data set gives rise to a phenomenon that we have called "the blessing of dimensionality", which will cause any search method to stumble upon a reasonable solution. The key is how to differentiate between various spurious solutions and a possibly true regulatory relationship. Since financial constraints make it very hard to conduct the experiments that would generate the amount of data necessary to find a unique optimally fitted network, this problem must be dealt with computationally.

Our approach is based on the assumption that genes that exhibit the same behavior are under the same kind of control. This assumption seems to be biologically sound, however, when we deal with a limited number of experiments, we may be fooled into believing that two genes that display the same behavior are part of the same regulatory pathway, since different strategies could generate the same response over a limited window of observation and under a limited set of conditions. Nevertheless, although this strategy could lead to false positives, it should not, in principle, hide any meaningful relationships.

Therefore, it was decided to use a clustering algorithm that assumes that genes that exhibit the same behavior are under the same kind of control, which means that strongly correlated genes should be treated together in order to find the single underlying regulation network that controls them all. It is important to understand that this clustering approach may not be correct, but searching for techniques that work in the entire cluster helps to eliminate many hypotheses and helps limit the number of control mechanisms found by the algorithm. Since the goal of this algorithm is to find testable hypotheses for the "wet lab", this helps to reduce the set of candidates.

One useful measure that indicates similarity between expression levels and their changes is the Pearson Product-Moment Correlation Coefficient (correlation coefficient for short) denoted ρ, which is a measure of the degree of linear relationship between two variables, usually labeled X and Y. While in regression the emphasis is on predicting one variable from the other, in correlation, on the other hand, the emphasis is on the degree to which a linear model may describe the relationship between two variables. In regression the interest is directional, one variable is predicted and the other is the predictor; in correlation the interest is non-directional, the linearity relationship is the critical aspect [12].

We calculate the correlation values for the gene of interest with every other gene in the microarray and establish an arbitrary cutoff point (in our case, we worked with a value of 0.95). Using only high correlation values is also interesting because it may generate overlapping clusters. The fact that gene g_1 is correlated to a certain degree with gene g_2 and the latter is correlated to the same certain degree with gene g_3 does not imply that gene g_1 is also correlated with gene g_3 to a degree above the cutting point. Therefore, we create one cluster for each gene of interest, containing all the elements that are correlated to it.

Pearson correlation values are in the range $[-1, 1]$, where -1 stands for perfect anti-correlated (when gene g_1 expression level rises, gene g_2 expression level lowers in the same proportion). We could include anti-correlation in our groups, just arranging for a special processing of the trees that turns inhibition into promotion and vice-versa, but we decided against it, because we do not need an additional hypothesis to constrain the veracity of our results.

10.4.3 Chromosome Structure

In our algorithm, a rule is represented as a tree whose linear representation is in reverse polish notation (RPN), where all operators precede their operands. This tree representation has already been used, although somewhat differently, in works such as [24, 8, 36].

In the model proposed, expressions are defined recursively by the following syntax:

$$\begin{cases} < Expression > ::= & \text{<Antecedent> <Consequent>;} \\ < Antecedent > ::= & \text{<BinOperator> <Expression> <Expression>} \\ < Antecedent > ::= & \text{NOT <Expression>;} \\ < Antecedent > ::= & \text{<FuzzySet>;} \\ < BinOperator > ::= & \text{AND | OR;} \\ < FuzzySet > ::= & \text{<set> <variable> <time>;} \\ < Consequent > ::= & \text{<set> <variable>.} \end{cases}$$

In this syntax, <set> defines the fuzzy set used, <variable> is one of the many genes whose expression levels were measured by the microarray and <time> is an integer that represents the delay between measurement and control. This delay is important because sometimes, due to kinetic energy in the molecules, probabilistic effects or mere reaction times, the control performed by one gene can occur later in time, especially if the interval between microarray measurements is small enough. This delay can go from 1 (previous time step) to $(n-1)$, where n is the number of measurements. It is important to understand that the number of time steps used to match the data will be limited by the maximum delay allowed (d). In this case, we will only have $n-d$ instances of data, starting at time step $n-d+1$.

The use of past data is important in several genetic processes. For instance, there is evidence that the phenomenon of cell memory is a prerequisite for the creation of organized tissues and for the maintenance of stably differentiated cell types [1]. In other cases, such as bacterial tryptophan control, only the immediate

availability of tryptophan is relevant. So, using the delay mechanism is a good addition to the toolbox but will not necessarily be used in all problems we might face.

This notation leads to a tree representation, where non-leaf nodes consist of operators, either binary (AND; OR) or unary (NOT) and leafs consist of the fuzzy set, the variable and the time delay. An example can be seen in Figure 10.1.

We must guarantee that we have at least one rule for each fuzzy set created for our gene of interest. We could consider having only one rule per gene and joining the various rules with an OR node, which would have the same effect in the defuzzification process. Nevertheless, this idea leads to higher trees and we choose to have several lower trees instead. This will cause only a minor impact on the application of the crossover operator (see Section 10.4.4 for further information). Therefore, our chromosome will consist of a "forest", with at least n_{fuzzy} trees, where n_{fuzzy} is the number of fuzzy sets created for our gene of interest.

Having more than one tree per fuzzy set can lead to having contradictory rules. Wang and Mendel [33] suggest having only one rule: the one with the maximum degree. We do not need to apply this rule because the OR interpretation in our algorithm will already do that, with the added benefit of being adaptive, meaning that in each different situation, a different rule may have the highest degree and will be chosen for application.

10.4.4 Genetic Operators

In order to define completely the proposed GP, we need to define the mutation and crossover operators. As is typical in evolutionary algorithms that use competing operators, the GP proposed in this paper uses a roulette wheel and assigns a time varying probability for the mutation and crossover operators.

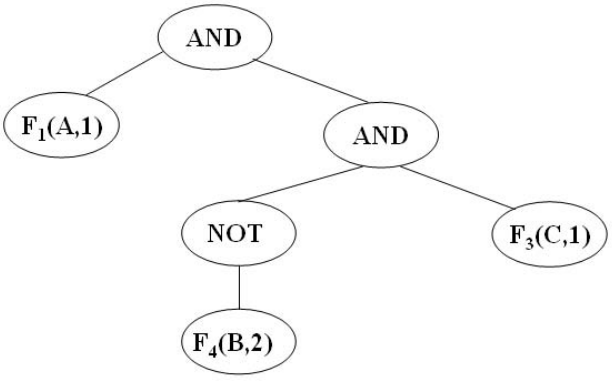


Fig. 10.1. Example of the representation. Inner nodes are the operators, while outer nodes represent a fuzzy set, a variable and a delay. In the figure, $F_1(A, 1)$ means that we are going to use the membership value of expression level of variable A in the previous time step (delay 1) to fuzzy set F_1.

The probability variation is due to the fact that in the beginning of each run we want to emphasize the exploitation aspect of the algorithm, by trying to allow the best characteristics in the population to propagate. This means that the crossover operator should have a higher choice probability. In the final generations, genetic convergence tends to occur and the population tends to be filled with similar individuals, which makes it more profitable for the GP to emphasize the exploration aspect, giving the mutation operator a higher probability.

Some researchers prefer to emphasize the exploration aspect in the early stages of the GA, but given the fact that the population is randomly initialized and applying further randomness on the population tends to yield little gain. On the other hand, giving the mutation operator higher probabilities in the later stages helps the GA fight the convergence effect that usually affects populations.

Therefore, we start with a high preference for the crossover operator (90% chance, for instance) and linearly decrease it with each successive generation. There are other strategies to make this chance (namely, quadratic and step decrease), but the choice of change mode does not seem to have a high impact on the GP performance.

Crossover Operator

The main goal of the crossover operator is to exchange information between two different individuals in a way similar to sexual reproduction, allowing the EA to exploit the best characteristics of the current population and hopefully transmitting it to the newly generated offspring.

The crossover operator used in this work is a version of the operator commonly used in genetic programming [19], having a performance that is quite similar to uniform crossover for genetic algorithms, both in its *modus operandi* and in the number of schemes preserved.

The operator works by randomly choosing sub-trees to interchange between the chosen parents, using the following algorithm:

```
For each parent do
    node n_c = tree root
    initialize selection probability p_s
    While n_c not equals to null do
        Make a random draw with probability p_s
        If n_c is chosen
            Store node n_c for the current parent
            Exit while
        Else
            raise probability p_s
        End if
    End While
End For
Create offspring by exchanging the selected sub-trees
```

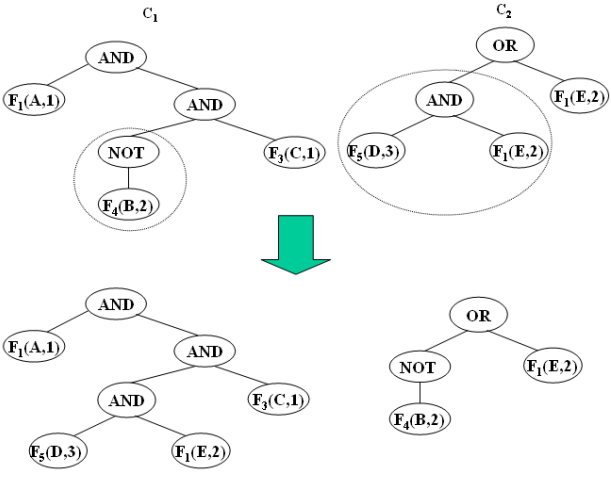


Fig. 10.2. Example of crossover action in two random parents

An example of this operator can be seen in Fig. 10.2. This strategy preserves the sub-expressions rooted at the selected nodes and therefore the crossover is not too disruptive with regard to the current population, which is a common problem with genetic programs.

Since there may be multiple rules per fuzzy set, there must be an additional control mechanism to choose which rules will exchange sub-trees. In this case, the crossover operator guarantees that the rules for a fuzzy set X in the first parent (C_1) only crosses with rules for the same fuzzy set in the second parent (C_2), even if there are more rules for this fuzzy set in one chromosome than in another. This means that if chromosome C_1 has two rules for fuzzy set X and chromosome C_2 has only one, the two rules from C_1 will cross with the single one from C_2. If the situation is reversed and C_1 has only one rule while has C_2 two or more, the number of rules in the resulting chromosome will still be equal to the number of rules in C_1.

If both of them have more than one rule for fuzzy set X, a random choice will be made between the alternative rules, only ensuring that each rule crosses at least once. Obviously, crossover generates two rules per operation, which are done applying the rules described here twice: first considering the order C_1/C_2 and secondly, the order C_1/C_2. Therefore, children with the characteristic structure of both parents will be generated.

We could create a multiple parent version of this crossover operator by allowing multiple cutting points in each parent and creating a pool of selected sub-trees that would be randomly selected to replace the cut branches of each parent. This version would be more disruptive and should be used together with an elitist operator and an increase in the number of generations per run.

Mutation Operator

The goal of the mutation operator is to insert random variation into the population so that we can explore areas of the search space that have not yet been visited. In order to perform this task, we create three different mutation operators:

1. rule mutation: The rule mutation randomly chooses one node in the rule tree and prunes the whole branch. Following this, a new sub-tree is generated using the same generator that created the initial population and the branch is then replaced. The population generator in this specific case is instructed to generate short trees (trees whose height is equal to or less than 3). An example of its operation is described in shown in Figure 10.3.
2. insertion mutation: Insertion mutation randomly chooses a fuzzy set for which to create a rule. The new rule is generated using the same random rule generator that generates the initial population. It is reasonable to give the inserted rules a high choice probability in this isolated creation, for we cannot raise the probability with the number of selections, as we did in the population initialization (see Section 10.4.7).
3. deletion mutation: Deletion mutation randomly chooses one rule to delete in one of the fuzzy sets available. In order to keep the rule base efficient, we cannot allow a fuzzy set to have zero rules, so the chosen rule will be deleted if it is not the only rule for a fuzzy set. In this case we will perform the

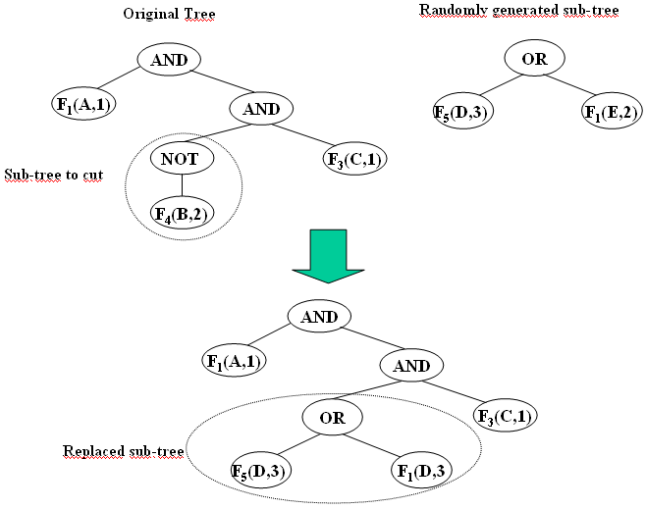


Fig. 10.3. The mutation operator chooses randomly a sub-tree to cut (circled) and replaces it with a newly generated short tree. The choice can be done algorithmically by performing a draw at each node with increasing probability. If the draw fails, we randomly choose to go to the left or right node and increase the probability in such a way that at a leaf, the draw probability will be 1.

random selection again, but one must be careful not to get into an infinite loop if all the fuzzy sets have exactly one rule, which can be avoided by allowing only a maximum number of tries for this operator. One must also be careful when excluding rules to verify whether the chosen tree is the only one that contains a rule inserted by the user.

10.4.5 Evaluation Function

In order to evaluate the performance of the chromosome, we will consider microarray data as a trajectory with N steps. This trajectory represents the "real" behavior of the network to be modeled, given the conditions it was submitted to. There are one or more genes of interest in that network, whose regulatory networks we intend to find.

In order to evaluate a proposed solution, the network it represents receives the first state of each trajectory and the GA calculates the intermediate and final steps for the genes of interest.

In order to calculate the expression levels at time t, one must decide whether to use the real or calculated values at each time step. As stated in the previous paragraph, the expression levels at time $t = 2$ are calculated using the real initial conditions at time $t = 1$ ($real_1$). But for every $t > 2$, how should we calculate the genes expression value ($calc_t$)? Should we use the real value at time $t - 1$ ($real_{t-1}$) or the previously calculated expression value at time $t - 1$ ($calc_{t-1}$)?

There are good arguments for both alternatives. Using the calculated values verifies if the network can model the entire trajectory, but allows errors to accumulate. Using the real values at each time point to calculate the next expression values verifies if the rules can, given a real value, predict the behavior of the network at the next point. Both are valid alternatives, but in our work we opted for the latter, because some experiments showed that some cumulative errors influenced the ability of the network to behave like the real one, specially when we use many fuzzy sets, causing them to have a small domain.

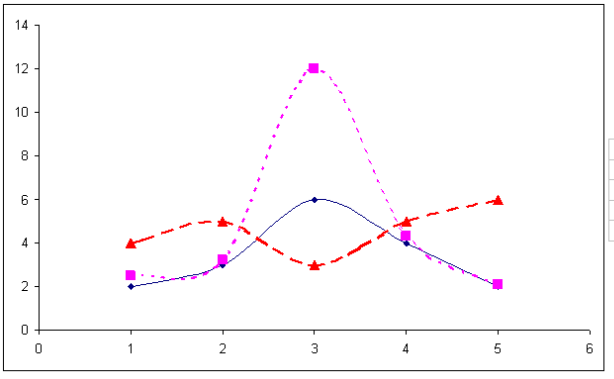
In order to avoid scale errors due to different expression levels at different time steps, the mean absolute percentage error (MAPE) was used, instead of using the absolute difference. This metric is defined by the following formula:

$$MAPE = \frac{1}{N} \sum_{t=1}^{N} \frac{real_t - calc_t}{real_t} \tag{10.2}$$

We will use the inverse of this function because the smaller the error, the bigger should be the evaluation function. We will also add 1 to the denominator in order to avoid infinity with perfect fit solutions. Therefore, the first try for our evaluation function will be:

$$F = \frac{1}{1 + MAPE} \tag{10.3}$$

The first impression is that this function will capture easily the good solutions, but there is a potential pitfall. In many situations a single outlier can make



x	real	dotted	traced
1	2	2,5	4
2	3	3,2	5
3	6	12	3
4	4	4,3	5
5	2	2,1	6

d(real, dotted)=36,39
d(real,traced)=34

Fig. 10.4. Example of the outlier effect. The solid line represents the real data and there are two calculated solutions represented by the dotted and the traced lines. Even though the dotted one models the underlying process better, it has an outlier at time t=3 that makes it MAPE grow and may cause it to be considered worse than the traced line whose behavior shows less resemblance to the real one.

the function obtain a low evaluation, even though it may correctly capture the changes in expression levels with time. An example of this problem is shown in Figure 10.4

In order to minimize this effect and reward those solutions that capture the directional changes in the target genes, we created a directional coefficient, which is given by the a function of the directional changes in the measured values ($real_t$) and the calculated values ($calc_t$), that is given by the following formula:

$$dc = \begin{cases} 0.7, \text{ if } (\uparrow real_t \wedge \downarrow calc_t) \vee (\downarrow real_t \wedge \uparrow calc_t); \\ 0.9, \text{ if } (\uparrow real_t \wedge \leftrightarrow calc_t) \vee (\downarrow real_t \wedge \leftrightarrow calc_t) \\ \quad \vee (\leftrightarrow real_t \wedge \downarrow calc_t) \vee (\leftrightarrow real_t \wedge \uparrow calc_t); \\ 1.1, \text{ if } (\downarrow real_t \wedge \downarrow calc_t) \vee (\uparrow real_t \wedge \uparrow calc_t). \end{cases} \tag{10.4}$$

In this formula, a $\uparrow$ means that the expression level has grown at time t. Conversely, a $\downarrow$ means that the expression level has diminished and a $\leftrightarrow$ means that the expression level has remained the same. An expression level is considered constant if it varies less than $\pm 1\%$ from the previous time point. This evaluation improves the capture of directional expressional movement in time. For instance, if an expression value increases from time t to time $t+1$ and the calculated value decreases in the same period of time, the error value at this time point may not be very high, even as a percentage. Therefore, on adding this coefficient, every chromosome that has a high evaluation will calculate a solution that has high correlation with the function that maps real expression values changes over time.

The blessing of dimensionality, discussed above, may cause the GP to find an over-specific rule that matches the numbers available without really uncovering the underlying process that generated them. This will be reflected in "tall" trees that should be discouraged by a reduction in their evaluation. This is achieved

by creating a coefficient $c \leq 1$ to multiply the evaluation function we discussed so far. The idea is that the higher the tree represented by the chromosome, the smaller is the coefficient c. Therefore, in the case where there are two different chromosomes with the same evaluation, the simpler and shorter one will be preferred. This does not imply that nature necessarily rewards simplicity, but rather that we have a small amount of data and should be cautious about over-fitting it. This coefficient is given by the following formula, dependent on the tree height h:

$$hc = \begin{cases} 1, & h \leq 2; \\ \frac{1}{h-1}, & h > 2. \end{cases} \tag{10.5}$$

The final formula is the multiplication of all three elements obtained in formulas 10.3, 10.4 and 10.5, creating the following evaluation function:

$$Eval(network) = hc * dc * F \tag{10.6}$$

This evaluation function is quite resilient and searches for the underlying process, not only achieving a data fitting, but also trying to avoid data over-fitting, by simplifying the regulatory process found and being aware of the dangers of the blessing of dimensionality.

10.4.6 Population Module and Execution Mechanism

In the GP proposed, we used an elitist population module and defined the following three different termination criteria:

- number of generations: no more than a predefined number of generations per run;
- stagnation: stop the run if the best solution stagnates for the last 20 generations. It was not used in our work, but an alternative solution is to increase the probability of the mutation operator, so that we give more emphasis to the exploration effect in order to find new solutions that break the stagnation. Of course, a fine control must be established so that the probability is decreased when the GP starts to have a better performance;
- quality of the solution found: stop if the data was fit to a maximum error of 1%. This number is arbitrary and appropriate for this specific problem. We know that the numbers obtained from microarrays are inherently imprecise (see Section 10.1.2) and that given the blessing of dimensionality, any perfect fit might consist of over-fitting. In other problems, the 1% rate of error might be too high;

Usually, genetic programs are seeded with large populations that execute for a large number of generations, because genetic programming operators tend to be very destructive, which causes a long execution time. The destructive effect can be minimized by using elitist strategies to preserve the best solutions and by using operators that prioritize the lower tree levels when deciding where to exchange material between genes.

In this work a different strategy was adopted. Ten independent runs with initial random populations were executed and the top 2% of each execution was used to seed the 11th run, whose initial population was completed randomly. This allowed for a faster execution time with smaller stagnation effects in each population, while allowing the best solutions found in different initial populations (seeded randomly) to interact in order to find a better solution. It is even possible to get an even bigger speed up by using a parallel execution of the algorithm, using 10 different machines that will allow their best solutions to migrate at the end of their execution.

10.4.7 Tree Processing

The rule generator used to initialize the population was created to guarantee that some chromosomes would incorporate prior knowledge, expressed as a specific set of rules defined by the user . This is an important advantage of fuzzy rules that was enforced in this work.

This incorporation was done by creating a random draw for each sub-tree generated to decide whether that sub-tree would be one of the rules inserted by the user. Since we wish for their value to be used as entered, placing those rules as descendants of a *NOT* node should be avoided. The probability assigned to this draw increased linearly with the number of chromosomes, up to the point where the rules were used or, in the final chromosome, the chance amounted to 100%, assuring rule usage.

A second passage was made through all generated chromosomes in order to assure that rules that were considered as "forbidden" by the user were absent from every chromosome. When cropping the tree to remove "forbidden" rules, a new sub-tree was generated with the characteristic discussed above.

After every reproduction/mutation cycle, a full pass through the new population is performed in order to verify that required rules are still present in the population and have not been disrupted by the genetic operators. If they are not present anymore, a new incorporation is performed, as discussed in the paragraph above.

A rule simplifier was created that allows us to substitute some sub-trees for simpler ones that still represent the same logical expression, allowing us to, for example, reduce expressions such as A AND A to their simpler form (in this case, A). The rules shown here have already gone through this simplification process. This simplification decreases the computational time spent by the proposed algorithm in the tree evaluation step, which is the longest one in the whole search process, decreases average tree height and also allows the genetic material exchanged between trees to be meaningful.

10.5 Results

The results obtained with the application of this algorithm to microarray data sets are described in great details in [23]. We will highlight their most important features in this section.

The data set tested was obtained from The Arabidopsis Internet Research project (TAIR) and measures the reaction to cold of *Arabidopsis thaliana*, giving the value of close to 8000 genes at seven data points. Because of this small number of measurements it was not feasible to use the time dependence available in the chromosomes. Therefore, every calculated value depends solely on the expression values available at the previous time point.

A. thaliana, like many plants, increases its freezing tolerance when exposed to low nonfreezing temperatures. This process of cold acclimation is a multigenic and quantitative trait that is associated with complex physiological and biochemical changes [17].

The genes that are hypothesized to be responsible for the cold response are 16062_s_at, 17520_s_at and 16111_f_at. The genes 13018_at and 13785_at are two of the genes regulated by the above named ones. The algorithm was applied to search for regulatory strategies for these last two genes, modeling correctly both trajectories and giving us the following interesting results:

- In the rules discovered for 13018_at, the three known regulators were present and a candidate regulator (17034_s_at) that was considered interesting enough by the biologists who provided the data to warrant further investigation in the near future. Other genes that show high correlation with known regulators were present in the rules, an effect that may be due to the small number of points available in the data set.
- In the second case (13785_at, some prior knowledge was included and the program was asked to include necessarily activation from 17520_s_at and preferentially an inhibition from 16111_s_at. The resulting rules included the required and the desirable relationships. Another interesting feature is the presence of gene 17034_s_at, which was also deemed interesting in the previous set of rules.
- In both cases, the rules present a few genes that don't seem to "belong" in terms of previous knowledge. This kind of spurious control relationship will always be present in any method and is a consequence of the blessing of dimensionality previously mentioned and cannot be avoided with such a small data set.

One can understand better those results by reviewing the best chromosome found for element 13875_at, which is described by the following rules:

(a) IF AND Low_Level_of_Expression(17413_s_at)
NOT High_Level_of_Expression(16111_f_at)
THEN Low_Level_of_Expression(13785_at)
(b) IF NOT Average_Level_of_Expression (15714_at)
THEN Low_Level_of_Expression(13785_at)
(c) IF NOT Average_Level_of_Expression (17834_at)
THEN Low_Level_of_Expression(13785_at)
(d) IF Low_Level_of_Expression (17421_s_at)
THEN Average_Level_of_Expression(13785_at)

(e) IF Average_Level_of_Expression (16062_s_at)
THEN Average_Level_of_Expression(13785_at)
(f) IF OR Low_Level_of_Expression(17050_s_at)
High_Level_of_Expression(16062_s_at)
THEN High_Level_of_Expression(13785_at)
(g) IF Average_Level_of_Expression (17034_s_at)
THEN High_Level_of_Expression(13785_at)
(h) IF NOT OR High_Level_of_Expression(16062_s_at)
Low_Level_of_Expression(15140_s_at)
THEN High_Level_of_Expression(13785_at)

Those rules have the following features worthy of note:

- Rules (a), (e) e (f) show relationships to known regulators;
- Rule (g) presents a new regulator (17034_s_at) that was considered promising by A. thaliana researchers;
- Rules (f) e (h) show relationships with elements 15140_s_at and 17050_s_at that are highly correlated to element 17520_s_at, which is a known regulator and is absent from the rules found. This situation is expected and is due to the small amount of data points.

10.6 Conclusion

The algorithm described was applied to a data set with many degrees of freedom and yielded interesting results. When applied to previously investigated regulation models, the results generated are very similar to known results in biology. This suggests that the algorithm may be a tool to uncover other regulation processes, but must be used with caution. All results found by this algorithm must be tested afterwards in a biological lab and all limitations associated with microarray data sets must be taken into consideration. Besides, if data is not scarce, other methods could be more effective, but, at the present time, obtaining a large number of microarray measurements is not financially feasible.

The approach described in this chapter to model gene expression is a very simplified view of the actual process, appropriate for exploratory data analysis. In reality, gene expression is a complex process regulated at several stages in the synthesis of proteins that also involves molecular movement and binding, which is a probabilistic event.

Apart from the regulation of DNA transcription, the best-studied form of regulation, the expression of a gene may be controlled during RNA processing and transport (in eukaryotes), RNA translation, and the post-translational modification of proteins. The degradation of proteins and intermediate RNA products can also be regulated in the cell, and the modeling of this process can be seen, for example, in [11]. Regulatory molecules can control the concentration and form of the product of each step. These regulators are usually fully-formed proteins, but any of the intermediate products (RNA, polypeptides, or proteins) also may act as regulators of gene expression. Reverse-engineering techniques

usually concentrate on protein transcription control, mainly because DNA microarray technology has become an abundant data source. Measuring peptide, protein and metabolite regulators of gene expression is generally more difficult, and such data are not often available [14].

In this situation, the model proposed here serves as a rough sketch of the regulatory process, but it still must be considered as an initial step and must be augmented in the direction of methods that can understand and model cellular context, RNA translation and protein folding; thus understanding the network as a whole.

It is important to understand that most large-scale data sets contain only information from cells exposed to a single condition. The approach proposed here does not attempt to analyze the dynamics of complex biological networks. In order to carry out such an analysis of dynamics, we would have to deal with more interaction data sets under different cellular conditions, and more importantly, integrate with gene expression profile data under various conditions. The reader interested in biological networks can refer to [37].

It is also important to understand that many different genetic networks can generate the same phenotype, specially under data scarcity conditions, a phenomenon called "gene elasticity" and discussed in detail in [20]. The approach proposed here will not find many different possibilities for the same data in the same run, but given the fact that there is a random initialization step in the GP, it is possible to find different networks in different runs.

Besides incorporating the issues described above, a full model must also incorporate different regulatory mechanisms at different time points. For instance, a gene may regulate another only at a certain time point, while remaining quiescent during the rest of the interval evaluated. A possible solution would be to add, to each fuzzy rule in the base, an application condition that would determine when to apply it. Many difficult issues arise from this idea: for example, ensuring that at least one condition applies to a controlled gene at every time step and determining how to submit these rules to a genetic operator. We are currently studying the best way to represent these application conditions in our rules.

References

1. Alberts, B., Johnson, A., Lewis, J., et al.: Molecular biology of the cell, 5th edn. Garland Science (2007)
2. Arnone, M.I., Davidson, E.H.: The hardwiring of development: organization and function of genomic regulatory systems. Development 124, 1851–1864 (1997)
3. Bansal, M., della Gatta, G., di Bernardo, D.: Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. Bioinformatics 22(7), 815–822 (2006)
4. Brazma, A., Rukliza, D., Viksna, J.: Reconstruction of gene regulatory networks under the finite state linear model. Genome Informatics 16(2), 225–236 (2005)
5. Carmona-Saez, P., Chagoyen, M., Rodriguez, R., et al.: Integrated analysis of gene expression by association rules discovery. BMC Bioinformatics 7(54) (2006)
6. Chen, J.J., Chen, C.-H.: Encyclopedia of biopharmaceutical statistics. In: Microarray Gene Expression, pp. 599–613. Informa Healthcare (2003)

7. Creighton, C., Hanash, S.: Mining gene expression databases for association rules. Bioinformatics 19(1), 79–86 (2003)
8. Dasgupta, D., Gomes, J.: Evolving fuzzy classifiers for intrusion detection. In: Proceedings of the 2002 IEEE Workshop on Information Assurance, US Military Academy (2002)
9. De Jong, H., Gouze, J.L., Hernandez, C., et al.: Qualitative simulation of genetic regulation models using piecewise-linear models. Bulletion of Mathematical Biology 66(2), 301–340 (2004)
10. Fogel, G.B., Corne, D.W.: Evolutionary computation in bioinformatics. Morgan Kaufmann, San Francisco (2003)
11. Foteinu, P., Yang, E., Saharidis, G.K., et al.: A mixed-integer optimization framework for the synthesis and analysis of regulatory networks. Journal of Global Optimization (online) (2007)
12. Freedman, D., Pisani, R., Purves, R.: Statistics, 4th edn. W. W. Norton Publisher (2007)
13. Freitas, A.A.: A survey of evolutionary algorithms for data mining and knowledge discovery. In: Ghosh, A., Tsutsui, S. (eds.) Advances in Evolutionary Computation, pp. 819–845. Springer, Heidelberg (2003)
14. Gardner, T.S., Fainth, J.J.: Reverse-engineering transcription control networks. Physics of Life Reviews 2(1), 65–88 (2005)
15. Gardner, T.S., Faith, J.J.: Reverse-engineering transcription control networks. Physics of Life Reviews 2(65), 88 (2005)
16. Georgii, E., Richter, L., Ruckert, U., Kramer, S.: Analyzing microarray data using quantitative association rules. Bioinformatics 21(suppl. 21), ii123–ii129 (2005)
17. Hannah, M.A., Heyer, A.G., Hincha, D.K.: A global survey of gene regulation during cold acclimation in Arabidopis thaliana. PLoS Genet. 1(2), 26–43 (2005)
18. Heaton, J.T.: Introduction to neural networks with java, 1st edn. Heaton Research, Inc. (2005)
19. Koza, J.R., Keane, M.A., Streeter, M.J., et al.: Genetic programming iv: Routine human-competitive machine intelligence (genetic programming), 1st edn. Springer, Heidelberg (2005)
20. Krishnan, A., Giuliani, A., Tomita, M.: Indeterminacy of reverse engineering of gene regulatory networks: The curse of gene elasticity. PLOS One (6), e652 (2007)
21. Laubenbacher, R., Stigler, B.: A computational algebra approach to the reverse engineering of gene regulatory networks. Journal of Theoretical Biology (229), 523–537 (2004)
22. Linden, R.: Algoritmos geneticos (genetic algorithms). Brasport (2006)
23. Linden, R., Bhaya, A.: Evolving fuzzy rules to model gene expression. BioSystems 88(1), 76–91 (2007)
24. Martinek, D.: A tree representation of fuzzy inference rules. In: Proceedings of 40th Spring International Conference MOSIS 2006, Prerov, CZ, p. 6 (2006)
25. Mistra, J., Schmitt, W., et al.: Iterative explorations of microarray gene expression patterns in a reduced dimensional space. Genome Research, 1112–1120 (2002)
26. Nuber, U. (ed.): DNA microarrays, 1st edn. Advanced Methods Series. Taylor and Francis, Inc., Abington (2005)
27. Pal, R., Datta, A., Bittner, M.L., Dougherty, E.: Intervention in context-sensitive probabilistic boolean networks. Bioinformatics 21(7), 1211–1218 (2005)
28. Ross, T.H.: Fuzzy logic with engineering applications. Wiley, Chichester (2004)
29. Schlitt, T., Brazma, A.: Current approaches to gene regulatory network modelling. BMC Bioinformatics 8(suppl. 6), 9 (2007)

30. Tegner, J., Yeung, M.K.S., Hasty, J., Collins, J.J.: Reverse engineering gene networks: Integrating genetic perturbations with dynamical modeling. PNAS 100(10), 5944–5949 (2003)
31. Van Someren, E.P., Wessels, L.F.A., Reinders, M.J.T.: Linear modeling of genetic networks from experimental data. In: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, pp. 355–366. AAAI Press, Menlo Park (2000)
32. Wahde, M., Hertz, J.: Coarse-grained reverse engineering of genetic regulatory networks. BioSystems 55(1), 129–136 (2000)
33. Wang, L., Mendel, J.M.: Generating fuzzy rules by learning from examples. IEEE Transactions on Systems, Man and Cybernetics 22(6), 1414–1427 (1992)
34. Wang, Y., Tetko, I.V., Hall, M.V., et al.: Gene selection from microarray data for cancer classification–a machine learning approach. Computational Biology and Chemistry 29(1), 37–46 (2005)
35. Wehrli, A.V., Grzegorczyk, M., Husmeier, D.: Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical gaussian models and bayesian networks. Bioinformatics 22(20), 2523–2531 (2006)
36. Yang, Z.R., Thomson, R., et al.: Searching for discrimination rules in protease proteolytic cleavage activity using genetic programming with a min-max scoring function. BioSystems 72, 159–176 (2003)
37. Zhu, X., Gerstein, M., Snyder, M.: Getting connected: analysis and principles of biological networks. Genes and Dev. 21, 1010–1024 (2007)

11

Infer Genetic/Transcriptional Regulatory Networks by Recognition of Microarray Gene Expression Patterns Using Adaptive Neuro-Fuzzy Inference Systems

Cheng-Long Chuang[1,2], Chung-Ming Chen[1], and Joe-Air Jiang[2]

[1] Institute of Biomedical Engineering,
National Taiwan University, Taipei, 106, Taiwan
clchuang@ieee.org
[2] Department of Bio-Industrial Mechatronics Engineering,
National Taiwan University, Taipei, 106, Taiwan

Summary. In our previous studies, we first noted that dot products of 1^{st}- and 2^{nd}-order differential derivatives are effective to extract paired expression patterns in time-course microarray gene expression data. Here, the feature extraction model and an adaptive neuro-fuzzy inference system are encompassed together for fusing information attained from microarray experiments and known gene-gene interactions confirmed by biological experiments. Having the capability allows computer to associate gene expression patterns with genetic or transcriptional interactions, we can use it to identify other interactions that have not yet been confirmed by biologists. We also bring forward the concept of multilayer adaptive neuro-fuzzy inference systems showing how fuzzy models collaborate with each other on the basis of limited time-course microarray data and take advantage of the known gene-gene interactions. In this chapter, we examine the fuzzy model that may lead to higher true-positive results in prediction of gene networks. We also discuss the relative merits and drawbacks of two methods, and provide recommendations for readers to choose the most suitable method specifically for their applications in bioinformatics.

11.1 Introduction

The genesis of bioinformatics is a discipline which makes the best effort of available computational resources in order to analyze the massive data sets produced by the modern technologies in molecular biology. In recently decades, bioinformatics has become an active research area in the fields of molecular biology, computer science, artificial intelligence, statistics and mathematics. Bioinformatics is a collaborative science that combines feature extraction, data analysis, database managing, and interpreting information attained from *in silico* molecular biology experiments. This research area has become one of the most cutting edge research topics after the Human Genome Project was completed in April 2003. With the improvements on microarray technology, abundant data in the field, such as gene sequences, protein sequences and structures, microarray gene

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 217–233.
springerlink.com

expression data, and results of polymerase chain reaction (PCR) experiments are exponentially growing. With the emergence of modern technologies, inferring genetic regulatory mechanisms in biological organism has become possible.

Facing the huge amount of data, traditional approaches are not applicable anymore since the biological problem itself is highly nonlinear. Also, it may cause computational overhead involved in computer implemented analysis systems. In order to search for answers the biological problems, advanced information technologies are needed. To answer these questions, the analysis algorithms needs to take advantage of large amount of data, and results in validly biological conclusions. Many computational and mathematical approaches had been published to make inferences about the molecular processes in animals, for example, data mining, graphical model, Bayesian learning, hidden Markov chain model, artificial neural networks, fuzzy logic, evolutionary optimization, and support vector machines. These approaches usually deal with three major problems in bioinformatics research, such as classification, clustering, and association.

Many algorithms have been proposed to predict regulatory networks, most of which using sequence data, localization data, expression data, structural data, or orthologs information across different species. Sequence data and expression data are the most frequently exploited biological features in most of the approaches. However, sequence based approaches do not provide a direct way to uncover the types of interactions, e.g. activation or repression, which constitute the actual functions of the interactions. Thus, expression data is required to compensate this drawback.

Recently, there have been a few studies on transcriptional compensation interactions. Transcriptional compensation is a phenomenon that the expression quantity for a gene increases when its compensatory gene is mutated. On the other hand, the expression quantity for a gene decreases when its compensatory gene is mutated, this is called transcriptional delimitation. Mechanisms of transcriptional compensation interactions are mostly discovered in quantitative real-time PCR (qRT-PCR) experiments. Transcriptional compensation and transcriptional delimitation are very useful clues to detect synthetic sick or lethal interactions, in which mutation of two or more genes causes more severe growth defection (i.e., symptoms resembling premature aging, Cockayne syndrome, etc.) than mutation of one gene alone. Therefore, inferring these interactions is an essential topic of interest in bioinformatics.

Fusing microarray gene expression data and prior knowledge about results attained in qRT-PCR experiments is still an open question for scientists to infer possible regulatory mechanisms of diseases in biological organisms. In this chapter, our focus here is on the task of answering question about how to combine graphical model and fuzzy logics, and use the hybrid model to infer genetic / transcriptional interactions. This task is arisen because large amount of microarray data and biological experiments results are now available for public access on the Internet. Hence, it should be more efficient if we simplify the problem of inferring genetic / transcriptional interactions into a machine learning problem. Our objective here is to provide a fuzzy logic based approach that enables the

machine to associate the known interactions with specific patterns in microarray gene expression data.

11.2 Genetic Interactions

As previously mentioned, genetic interactions can be uncovered by qRT-PCR experiments. qRT-PCR system is a successful and well-known tool that enables reliable measurements for establishing functional linkages between a group of genes. qRT-PCR system achieves quantitative PCR analysis by taking advantage of the specific interaction between two modified genes sequences. One of the PCR primers contains a fluorescent label that residues at the 5'-terminus. The other PCR primer is unlabeled. The reaction mix includes both primers, and the labeled primer is preferentially incorporated at the position complementary to the target double-stranded DNA. The incorporation of the unlabeled strand at this position results in quenching of the fluorescent dye on the complementary DNA strand and a reduction in fluorescence, which allows quantization during amplification. Multiplex reactions are achievable by using different fluorescence on a primer pair for each target sequence. After the amplification procedures are completed, the labeled strands fluoresce when hybridized with a complementary DNA strand. The measured fluorescence intensity does not have any absolute unit associated with it. Thus, it usually gives ratio of the fluorescence intensities comparing between different tissues or experimental conditions. In this experiment, we can attain the answer of: Does the expression level of gene of interest change due to an experimental condition?

After a series of qRT-PCR experiments, we measure the fluorescence intensity levels of genes of interest. Furthermore, statistical test is conducted to check the functional linkage between a gene pair, say gene A and gene B, where A is a potential regulating gene, and B is a potential target gene regulated by A. Two groups of qRT-PCR experiments are conducted in which one is treatment group, and the other is control group. First, we measure the level of B against that of its partner mutative gene A. We also include wild type gene A as the control group. Each group contains a number of replicates, say n replicates. According to our biology knowledge, if A functions as an activator of B, the level of B should decrease when $A$was mutated. On the other hand, if A functions as an repressor of B, the level of B should increase when A was mutated.

In order to verify the aforementioned activator-to-target interaction between A and B, fluorescence intensity levels of B in all experiments are measured, where $C_1, C_2, \ldots, C_{n1}$ are levels of B in control group (wild type gene A), and $E_1, E_2, \ldots, E_{n2}$ are levels of B in experimental group (mutative gene A). We test the hypothesis using t-test:

$$\begin{cases} H_0 : \mu_C = \mu_E \\ H_1 : \mu_C \neq \mu_E \end{cases},$$

where μ_C and μ_E is the mean of levels of B in control group and experimental group, respectively, and significance level $\alpha = 0.1$. Since the numbers of replicates

in control group and experimental group are not necessarily to be the same, we utilize independent two-sample t-test with unequal sample sizes and unequal variances to test the hypothesis. The$t$ statistic to test whether the means of fluorescence intensity levels of B in two groups are different can be calculated by:

$$t = \frac{\bar{C} - \bar{E}}{s_{\bar{C}-\bar{E}}},$$

where

$$s_{\bar{C}-\bar{E}} = \sqrt{\frac{s_C^2}{n_1} + \frac{s_E^2}{n_2}},$$

and s_C^2 and s_E^2 are the unbiased estimators of the variances of the levels of B. For the significance testing, the distribution of the test statistic is approximated by the ordinary Student's $t$distribution. The degree of freedom of the Student's $t$distribution can be determined by the Welch-Satterthwaite equation, which can be formulated by:

$$df = \frac{\left(\frac{s_C^2}{n_1} + \frac{s_E^2}{n_2}\right)}{\frac{\left(\frac{s_C^2}{n_1}\right)^2}{n_1 - 1} + \frac{\left(\frac{s_E^2}{n_2}\right)^2}{n_2 - 1}}$$

Since it is a two-tailed test, the null hypothesis (H_0) is rejected if the p-value of the test is lesser than or equal to $\alpha/2 = 0.05$ on either side of the Student's $t$distribution. Thus, the observation is inconsistent with H_0, which means that the levels of B in control group and experimental group are significantly different due to A was mutated.

On the occasion that H_0 is rejected, a functional linkage is confirmed to exist between the regulating gene A and the target gene B. As aforementioned in the first section, we focus on two types of interactions: transcriptional compensation and transcriptional delimitation. If μ_C is lesser than μ_E, it represents that mutative gene A increases the level of B, and the interaction between A and B is transcriptional compensation interaction. Alternatively, If μ_C is greater than μ_E, it represents that mutative gene A decreases the level of B, thus the interaction between them is transcriptional delimitation interaction. A set of prior knowledge about existing interactions can be attained by conducting a series of qRT-PCR experiments and statistical tests. The prior knowledge data set is then used to train the fuzzy inference system that will be introduced later.

11.3 Time-Course Microarray Gene Expression Patterns

In the field of molecular biology, gene expression levels of thousands of genes can be measured at once using modern microarray technology. A measurement represents a global picture of genomic functions at a particular time. Many experiments of this sort measure the activities of genes over a period of time, and result in time-course gene expression data. Time-course gene expression data

is inheritable information that determines the cumulative amount of messenger RNA (mRNA) made from a gene. It can have a profound effect on the functions of the gene in the organism. If a gene is used to produce mRNA, it is considered to be activated, otherwise, it is repressed. Thus, gene expression patterns between genes sometimes been interpreted as a kind of gene-gene interaction process.

There are many kinds of metric that have been proposed to measure the similarity of gene expression patterns between genes, such as Euclidean metric or manifold metric. Pearson correlation is also a widely utilized method to measure the similarity of two genes' expression pattern. However, according to our knowledge in molecular biology, any particular promoter region in the upstream sequence of a gene can be bound by both its activators and repressors, and the activators or repressors might also require co-factors to enable its function. Hence, it leads the gene regulating model to become a highly nonlinear system, which is capable of responding in multiple ways to a change of gene activity depending on the state of the overall gene regulatory network and the presence of genes affecting different pathways. For this reason, traditional linear metrics are not suitable to analyze patterns in microarray gene expression data.

In this chapter, we utilized a curve feature based metric to measure the similarity or counter-similarity of paired genes' expression levels as originally proposed in PARE by Chuang, *et al.* [2]. This metric was originally inspired by observing the patterns in the plots of paired genes' expression levels. By comparing the observed patterns with interactions that were confirmed by qRT-PCR experiments, two major types of patterns were found. The first type of the patterns is similar pattern, and the second type is anti-similar pattern. In [2], both patterns have been checked against interactions (e.g. transcriptional compensation and transcriptional delimitation) that have been confirmed by qRT-PCR experiments. A t-test was conducted to verify the significant association between patterns and interactions. With the significant level of 0.05, transcriptional compensation interactions were identified to have anti-similar pattern. As well, transcriptional delimitation interactions were identified to have similar pattern in microarray gene expression data. An example of transcriptional compensation interaction SWE1-HST3 in the plot of expression levels is given in Figure 11.1(a). We can see that the expression levels of both genes show the anti-similar pattern. Another example, as depicted in Figure 11.1(b), shows a transcriptional delimitation interaction POL32-TOP1. It is clear to observe a similar pattern in the plot of both genes' expression levels.

According to our observation on example depicted in Figure 11.1(a), the slopes and curvatures of gene expression curves of similar pattern tend to have the same signs during the experimental period. Also, the example plotted in Figure 11.1(b) shows that the slopes and curvatures of gene expression curves of anti-similar pattern tend to have different signs for the whole experimental time course. To capture the characteristics of similar pattern and anti-similar pattern of expression curves, we calculate the products of their first and second derivatives with respect to time. We formulate these into two functions at below:

$$S_{A,B} = \frac{\partial G_A(t)}{\partial t} \cdot \frac{\partial G_B(t)}{\partial t},$$

$$C_{A,B} = \frac{\partial^2 G_A(t)}{\partial^2 t} \cdot \frac{\partial^2 G_B(t)}{\partial^2 t},$$

where $S_{A,B}$ and $C_{A,B}$ represents the products of the first derivatives and second derivatives of expression levels for regulator gene A and target gene B, respectively. $G_A(t)$ and $G_B(t)$ are gene expression level of A and B at time point t. The partial differential terms need not to be continuous there. Since time-course gene expression levels are discretely measured in time, so the 1^{st}- and 2^{nd}-order partial differential terms for computing $S_{A,B}$ and $C_{A,B}$ can be formulated by

$$\frac{\partial G(t)}{\partial t} = \frac{G(t+1) - G(t)}{\Delta t},$$

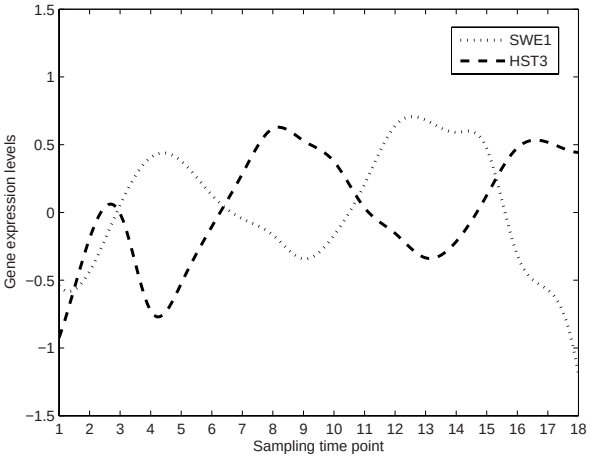
$$\frac{\partial^2 G(t)}{\partial^2 t} = \frac{G(t+2) - G(t+1)}{\Delta t \Delta(t+1)} - \frac{G(t+1) - G(t)}{(\Delta t)^2},$$

$$= \frac{G(t+2) - 2G(t+1) + G(t)}{(\Delta t)^2}$$

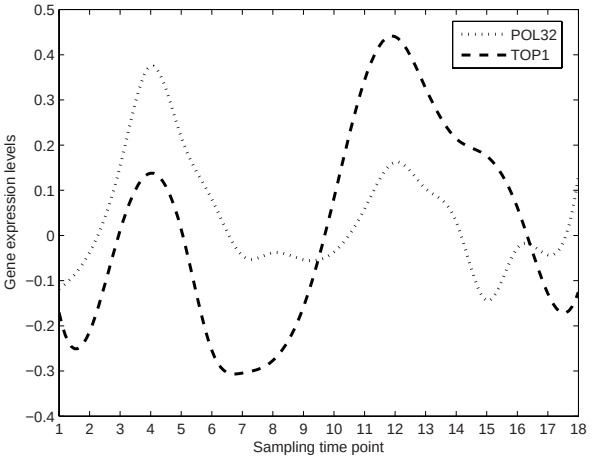
Paired gene expression levels with similar pattern will result in positive $S_{A,B}$ and $C_{A,B}$. Contrarily, anti-similar pattern will result in negative $S_{A,B}$ and $C_{A,B}$. These two formulas have been verified to be effective to extract nonlinear features of similar pattern and anti-similar pattern in microarray gene expression data.

As aforementioned, coefficient of Pearson correlation has been widely applied to estimate the strength of trend between two random variables. In the field of bioinformatics, gene expression levels are equivalent to random variables in statistics. Thus, many researchers used Pearson correlation to determine the strength of functional linkage between two genes. However, microarray gene expression data is known to be nonlinear, and it is necessary to do some modifications on Pearson correlation in order to make it suitable for analysis of microarray gene expression data. Based on our knowledge in signal processing, if a set of signal is nonlinear and discontinuous, ordinary signal processing methods might become inapplicable. But if we divide the signal into several smaller fragments, the nonlinearity and discontinuity of the signal can be removed, and those signal processing methods are then become applicable. Base on this concept, we introduce another approach, named "sum of local correlation coefficient (SLCC)", to measure the nonlinear correlation between two random variables. It combines much of the simplicity of linear correlation with the flexibility of measuring nonlinear correlation. A moving window is needed to compute local correlations denoted by r'. If the data comes from two gene expression levels $G_A(t)$ and $G_B(t)$, then SLCC is formulated by average of local correlations as follow:

$$r_{A,B} = \frac{1}{n-w} \sum_{i=1}^{n-w} r' \left(\left[G_A(i) \cdots G_A(i+w-1) \right], \left[G_B(i) \cdots G_B(i+w-1) \right] \right),$$



(a)



(b)

Fig. 11.1. (a) The gene expression pattern of transcriptional compensatory (TC) gene pairs SWE1 and HST3 across time; (b) The gene expression pattern of transcriptional diminished (TD) gene pairs POL32 and TOP1 across time

where $r_{A,B}$ is the SLCC between A and B, n is the total number of time points, and w is the size of the moving window. The local correlations r' is defined by products of the standard scores of the two gene expression levels x and y divided by the degrees of freedom

$$r'(x,y) = \frac{1}{w-1}\sum_{i=1}^{w}\left(\frac{x_i-\bar{x}}{s_x}\right)\left(\frac{y_i-\bar{y}}{s_y}\right),$$

where $\bar{x}$ and s_x represent sample mean and sample standard deviation of x, respectively. Similar to $S_{A,B}$ and $C_{A,B}$, gene expression levels belong to similar pattern and anti-similar pattern will result in positive and negative r, respectively. The size of the moving window w is suggested to be one half time of cell cycle.

Please note that $S_{A,B}$, $C_{A,B}$, and r aforementioned in above are based on the assumption that it has no time-lag effect on genetic interactions. Some previously studies (e.g. [5]) suggested that co-expressed genes usually do not regulate the functions of one another. Therefore, observing gene expression data taken with time lags of one or two time point might be useful to capture the causal effects in which the expression behavior of one gene leads to a delayed pattern of expression of another [9, 2]. Also, both biological cellular behavior and the time interval between two time points should be taken into account when deciding the length of the time-lag. If the time interval is short (e.g. minutes), we suggest that taking one or two time-lags is reasonable. Alternatively, if the time interval is long (e.g. days), no time-lag is needed during the analysis.

11.4 ANFIS: Adaptive Neuro-Fuzzy Inference Systems

Fuzzy inference is a computer paradigm based on fuzzy set theory, fuzzy rule base, and fuzzy reasoning. With the concept of uncertainty, a fundamental fuzzy system was introduced by Zadeh's paper in 1976. The significance of Zadeh's paper was that it was designed to mathematically represent uncertainty and vagueness based on the foundation of probability theory. The capability of fuzzy sets is to express gradual transitions from membership function to non-membership function and vice versa. There are a number of ways that can involve fuzzy sets in a system, such as analogous text description of the system, uncertainty expression in system parameters, and inputs, outputs and state variables that are described by fuzzy sets. Different from crisp systems, fuzzy systems were modeled by means of if-then rules. While the text description is vague and less specific, it is usually more useful to describe a system that a mathematical model is difficult to derive. Also, it allows decision making with estimated value under incomplete or uncertain information.

In general, a fuzzy inference system consists of four primary parts: fuzzification, inference, fuzzy rule base, and defuzzification. The overall block diagram of the fuzzy inference system is depicted in Figure 11.2. Depending on the form of formulation, two types of fuzzy inference models can be identified. The first type is Linguistic fuzzy model, where both the antecedent and the consequent are fuzzy propositions. The other one is Takagi-Sugeno fuzzy model, where the antecedent is fuzzy proposition, but the consequent is a crisp function. The knowledge represented by fuzzy rules can be easily understood and handled by human being. For a thorough review of fuzzy systems, we refer to [15, 16, 17, 7, 13].

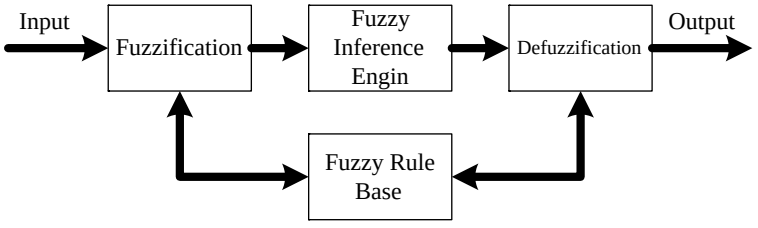


Fig. 11.2. General model of an adaptive neuro-fuzzy inference system

Fuzzy inference systems have been proven to work well in reasoning with imprecise information. Although fuzzy logic can be used to encode expert knowledge directly using linguistic rules, it usually takes a lot of time to design and tune the membership functions which quantitatively define those linguistic rules. In some nonlinear applications, the complexity in developing fuzzy rules increases with complexity of the entire fuzzy inference process, such as selection of membership functions, configuration of the center and width of the membership functions, and development of the appropriate fuzzy rules. Moreover, it is very difficult for designers to express the nonlinear fuzzy rules. Membership functions and fuzzy rules are not aggregated completely with easily without a systematic and reliable manner. Thus, these disadvantages make the applications of fuzzy systems been restricted to the field where expert knowledge is available and the number of input and output variables is small. Due to they cannot automatically acquire the rules they use to interpret those information, it might make the fuzzy inference systems unsuited for some particular problems. The limitation of learning is therefore of interests in the development of fuzzy systems, and it has become a central driving force behind the creation of the adaptive neuro-fuzzy inference systems (ANFIS).

ANFIS, which was introduced in [4], is a hybrid system that incorporates the advantages of fuzzy inference systems and neural networks whose interconnection weights are fuzzy sets. It combines the capability of fuzzy sets theory in handling uncertainty information and the ability of neural networks in learning from a set of training data through numbers of trials. The combination of these particular features makes ANFIS to have the advantages of adaptability and fault tolerance on both linear and nonlinear problems.

The architecture of the neuro-fuzzy model is shown in Figure 11.3. The architecture of ANFIS consists of five layers feedforward network, which is comprised by the input, defuzzification, rule, composite, and output layer:

In the 1^{st} layer, the output of every node is the degree of association between input and the linguistic label associated with this node function. The membership function of each node specifies the degree to which the given input satisfies the linguistic label. The output of i^{th} node in 1^{st} layer $O_{1,i}$ can be formulated by

$$O_{1,i}(x) = \frac{1}{1 + \left(\left(\frac{x - c_i}{a_i} \right)^2 \right)^{b_i}},$$

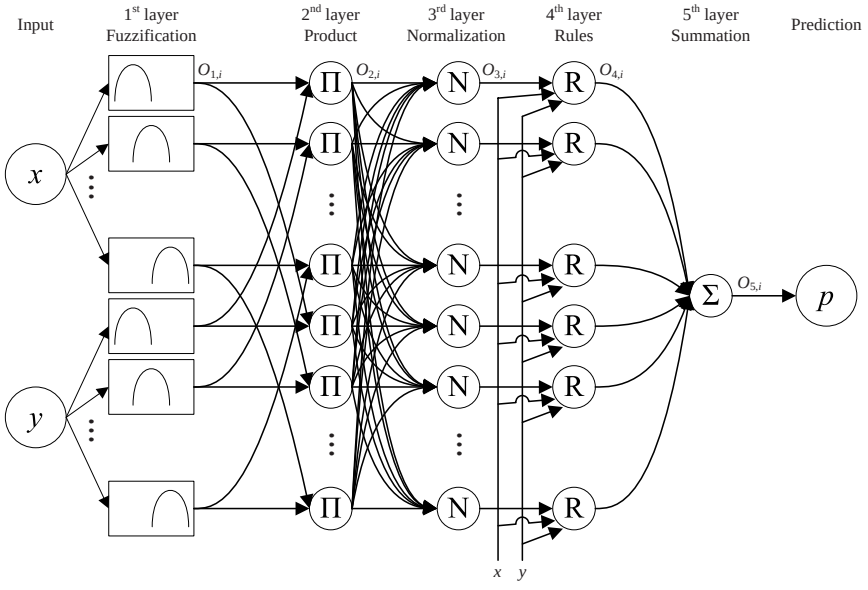


Fig. 11.3. The conceptual diagram of ANFIS, which has two inputs, one output, and three member functions (CP, SP and no pattern) corresponding to each input

where x is the input of the i^{th} node, and $\{a_i, b_i, c_i\}$ are parameter set need to be tuned. The values of these parameters determine the shape of membership function on linguistic label of i^{th} node. The function of $O_{1,i}$ can be bell-shaped function, or other continuous functions such as trapezoidal or triangular-shaped functions.

Every node in the 2^{nd} layer multiplies the incoming signals and sends the product out to the next layer. The operation in the 2^{nd} layer can be formulated by

$$O_{2,i}(x) = \prod_i O_{1,i}(x).$$

The value of $O_{2,i}$ represents the firing strength of a rule. It is equivalent to fuzzy AND operation of inputs from the 1^{st} layer.

The node in the 3^{rd} layer calculates the ratio of the i^{th} rule's firing strength to the sum of all rules' firing strengths, which can be formulated by

$$O_{3,i}(x) = \frac{O_{2,i}(x)}{\sum_j O_{2,j}(x)}.$$

The output of every node is a value with maximum equal to 1 and minimum equal to 0. Thus, output if this layer is also called normalized firing strength.

In the 4^{th} layer, the output of i^{th} node is defined as below

$$O_{4,i}(x) = O_{3,i}(x) \cdot f_j,$$

where f_j is the function of rule defined in rule base. The function f_j may contain several parameters that need to be estimated.

The node in the 5^{th} layer calculates the overall output of the ANFIS as formulated at below

$$O_{5,i}(x) = \sum_i O_{4,i}(x).$$

The operation of node in 5^{th} layer is the summation of all incoming signals from the 4^{th} layer. Total number of rules in ANFIS can be derived by product of numbers of membership functions associated with every input.

A numerous type of learning algorithms can be employed to update the aforementioned parameters. In the forward process of the training, parameters in f_j are estimated by the least squares estimate. In the backward process, the error rates propagate backward to update parameter sets (e.g. $\{a_i, b_i, c_i\}$ in the 1^{st} layer) by gradient descent. In order to reduce the computational complexity of the training process, some studies used gradient descent only or gradient descent with one pass of least square estimate to update all parameters. If we utilize gradient descent in larger portion of the training algorithm, the computational time can be significantly reduced. However, it might also reduce the resulting performance of the training. Various types of training algorithm have been developed to attain optimal values for parameters in ANFIS. For a thorough review on other training algorithms, please refer to [14, 6, 1, 3, 8].

11.5 Predicting Genetic Interactions Using Multilayer ANFIS

By combining the feature extraction method in Section 11.3 and ANFIS in Section 11.4, a new algorithm, called GeneCFE-ANFIS, is developed to infer genetic interactions using microarray gene expression data. The architecture of the GeneCFE-ANFIS is depicted in Figure 11.4. First, the raw data measured from microarray experiments is normalized. The purpose of normalization is to adjust microarray data for noises which added by variation in the technology or human error rather than from biological difference between the mRNA samples [11]. The association between dye-bias and fluorescence intensity can be observed by plotting an MA-plot. If the MA-plot shows a skewed trend, the experiments might be considered to have dye-bias, and it might lead the analysis to omit some spots that might be differentially expressed genes of interest. In order to remove the dye-bias effect from raw data, an overall trend line is estimated by loess regression, and corrects the M-values in the MA-plot by subtracting the values estimated by loess regression from the M-values. The global loess normalization can be formulated as below

$$N = M - loess(A),$$

where $M = \log_2 R - \log_2 G$, $A = (\log_2 R + \log_2 G)/2$, N is the normalized log-ratio intensity for each spot, and R and G represent red and greed intensities

measured from each spot, respectively. Sometimes the difference between each print-tip might also lead the data to suffer from print-tip bias. Print-tip loess normalization is then applied in order to adjust the log-ratio within each print tip group. The process of the print-tip loess normalization is similar to the global loess normalization. For more information about normalization of microarray data, please refer to [11]. Except dye-bias and print-tip-bias, time-course microarray gene expression data are known to suffer from severe noise problem because the data are measured from multiple microarray experiments. Noise signals embedded in the data usually seriously influence the performance of the analysis. In order to remove noise signals from the data, a mean filter is applied to the normalized log-ratio data. The size of the mean filter depends on the strength of noise signals. In Chuang *et al.* [2], mean filter has been proven to be effective to help discovering the macro trend of the gene expression patterns.

Then, we enter the feature extraction stage of GeneCFE-ANFIS. The GeneCFE-ANFIS is an algorithm that predicts genetic / transcriptional interaction in pairwise manner. For example, if we are looking for the effect of regulating gene A acts on its target gene B, the gene expression levels of A and B are applied to the feature extraction method aforementioned in Section 11.3. We can obtain a feature vector $\{S_{A,B}, C_{A,B}, r_{A,B}\}$ that identifies the pattern formed by gene expression levels of A and B, and the feature extraction stage is complete.

In the inference stage of GeneCFE-ANFIS, the feature vector $\{S_{A,B}, C_{A,B}, r_{A,B}\}$ is applied to a two-layer ANFIS to infer the interaction between A and B. As shown in Figure 4, the front-end and back-end ANFIS are encompassed by three and one ANFIS, respectively. Elements of feature vector $S_{A,B}$, $C_{A,B}$,$r_{A,B}$ are assigned to signal-input and single-output ANFIS-D1, ANFIS-D2, and ANFIS-Corr in the front-end ANFIS, respectively. The back-end ANFIS is a three-input single-output ANFIS. The input signals of the back-end ANFIS are the output signals from the front-end ANFIS. The output of the back-end ANFIS represents the prediction result. The outputs of all ANFIS in both layers can be interpreted by: 1) positive value represents activator-target or transcriptional delimitation interaction, and 2) negative value represents repressor-target or transcriptional compensation interaction.

The purpose of the two-layer ANFIS design is a realization of voting scheme. The inference results yielded by front-end ANFIS are preliminary predictions based on each element in feature vector $\{S_{A,B}, C_{A,B}, r_{A,B}\}$, individually. These preliminary predictions are then sent to the back-end ANFIS to obtain a score of final prediction. This voting scheme helps the algorithm to isolate the low quality element in the feature vector from the analysis. The low quality element means that the feature extracted from paired genes' expression pattern does not show much association with its interaction. This situation is normal and happens all the time because the characteristics of microarray gene expression data usually differ from one another. The two-layer ANFIS design can avoid such problem during the training process, and the overall prediction accuracy is better than that uses single ANFIS only.

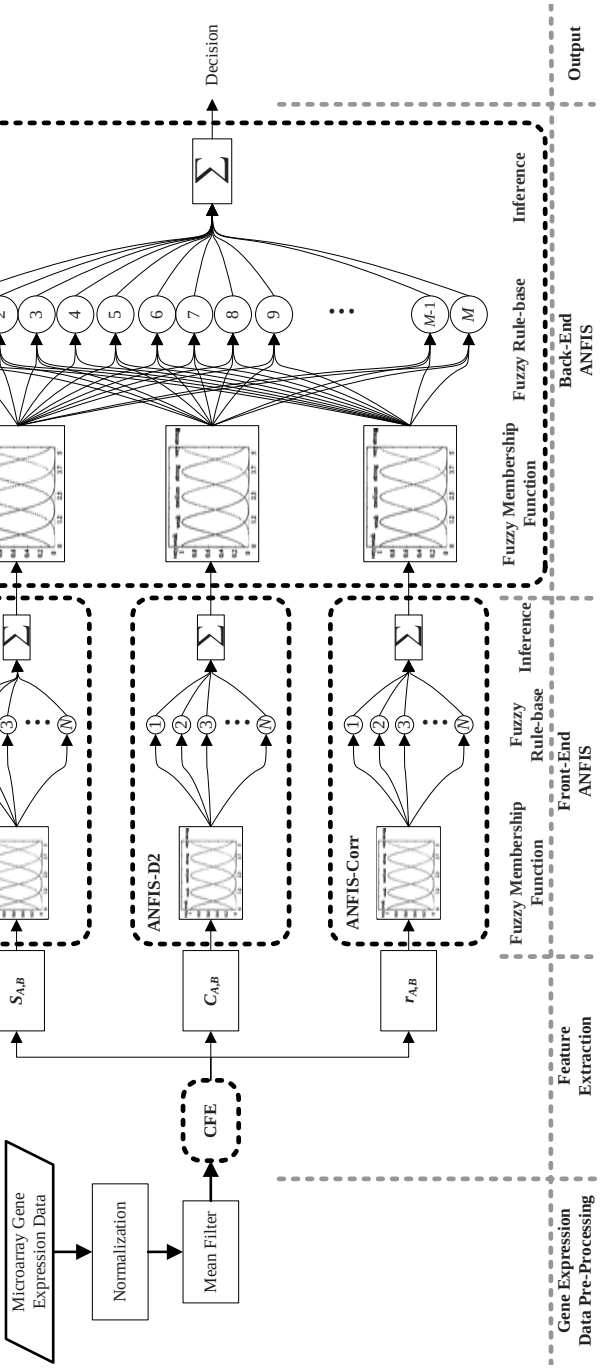


Fig. 11.4. Architecture of the proposed GeneCFE-ANFIS, which consists of several procedures: normalization, mean filter, features extraction using CFE, and fuzzy inference on gene-gene interactions cooperating with ANFIS

11.6 Experimental Results

In this chapter, the GeneCFE-ANFIS is applied to a set of cDNA microarray data that was provided by Spellman *et al.* [12] to infer potential genetic / transcriptional interactions. The raw data of the microarray data set is available online for public access at http://cellcycle-www.stanford.edu. The microarray data set encompasses four sub data sets labeled by alpha, cdc15, cdc28, and elu. The alpha data includes 18 time points following removal of alpha factor added 120 minutes earlier. The cdc15 data set consists of 25 expression level series from 23 time points following arrest of cdc15 temperature sensitive mutant. The cdc28 data set contains 17 time points following arrest of cdc28 temperature sensitive mutant. Finally, the elu data set is formed by 14 time points following elutriation. For each experiment, experimental and control groups were fluorescence intensities measured from synchronized yeast cultured by its experimental conditions. Log ratio of red to green fluorescence intensities were applied to the GeneCFE-ANFIS for the reconstruction of the genetic regulatory networks.

Since GeneCFE-ANFIS requires a complete training before it can produce any useful prediction, a training data set formed by known interactions is needed. A total of 112 genetic interactions (88 transcriptional compensation interactions and 24 transcriptional delimitation interactons) that were confirmed by qRT-PCR experiments are utilized to build a prior knowledge database. Among these prior known gene interactions, $(1/k)$ of interactions were randomly chosen to be a training set to tune the parameters of GeneCFE-ANFIS, and the rest $1/k$ portions of interactions forms the test set for k-fold cross validation (CV), where $k = 3$, 10 and 112 (leave-one-out CV) were implemented. Note that 3-fold, 10-fold and n-fold CVs were also conducted for the training sets to obtain the averaged accuracy, which can be checked against the true-positive rate to detect any overfitting problem. GeneCFE-ANFIS is applied to the entire database consisting of alpha, cdc15, cdc28 and elu data set provided in [12]. The simulation results are summarized in Table 11.1. The true-positive rate is defined to be the ratio of the correctly predicted pairs to the total number of gene pairs. The best case was obtained in elu data set, which was checked against the well-known interactions confirmed by qRT-PCR. The true-positive rates of GeneCFE-ANFIS with 3-fold, 10-fold, and n-fold CVs in elu data set are 69%, 70%, and 79%, respectively, where 3-fold and 10-fold CV were repeated for 500 times.

A total of 77 transcriptional interactions (56 repressor-target interactions, 21 activator-target interactions) that were collected by surveying previously published literatures are utilized as knowledge database to train and test the performance of GeneCFE-ANFIS. Among these prior known gene interactions, $(1/k)$ of interactions were randomly chosen to be a training set to tune the parameters of GeneCFE-ANFIS, and the rest portions $1/k$ of interactions forms the test set for k-fold cross validation (CV), where $k = 3$, 10 and 77 (leave-one-out CV) were implemented. GeneCFE-ANFIS is applied to predict interaction relationships of the collected TFs pairs using all data sets in [12]. The simulation results are summarized in Table 11.2. True-positive rates of GeneCFE-ANFIS in each

Table 11.1. The prediction results of GeneCFE-ANFIS applied to the *alpha*, *cdc15*, *cdc28* and *elu* data sets for all time points. The simulation results were checked against the 112 pairs of gene interactions confirmed by qRT-PCR experiments.

Dataset	CV Type	Training	Testing
		Training Accuracy	True-Positive Rate
Alpha	3-fold	86.9%	60.3%
	10-fold	81.4%	61.5%
	n-fold	79.6%	62.5%
Cdc15	3-fold	89.5%	64.7%
	10-fold	86.9%	65.0%
	n-fold	85.6%	66.9%
Cdc28	3-fold	89.1%	61.1%
	10-fold	86.9%	63.9%
	n-fold	86.1%	64.3%
Elu	3-fold	89.3%	69.2%
	10-fold	88.9%	70.4%
	n-fold	87.8%	79.5%

Table 11.2. The prediction results in terms of accuracy and true-positive rate (TPR) of GeneCFE-ANFIS applied to the *alpha*, *cdc15*, *cdc28* and *elu* data sets with all time points. The simulation results were checked against the 77 TFs pairs collected from literatures.

Dataset	CV Type	Training	Testing	Schäfer and Strimmer [10]
		Accuracy	TPR	TPR
Alpha	3-fold	85.5%	59.3%	52%
	10-fold	79.7%	60.5%	
	n-fold	79.6%	66.2%	
Cdc15	3-fold	87.8%	59.8%	52%
	10-fold	83.4%	61.2%	
	n-fold	81.1%	62.3%	
Cdc28	3-fold	85.5%	60.2%	56.9%
	10-fold	83.4%	62.7%	
	n-fold	82.9%	63.6%	
Elu	3-fold	90.5%	63.3%	57.9%
	10-fold	86.2%	69.2%	
	n-fold	85.3%	71.4%	

data set range from 59% to 71%, which are checked against 77 transcriptional interactions with the known interaction relationships. The same simulations based on the model proposed in [10] were also conducted, and the true-positive rate are 52%, 52%, 56.9%, and 57.9% for alpha, cdc15, cdc28, and elu data sets, respectively. The results show that GeneCFE-ANFIS overcomes the performance yielded by the model proposed in [10], especially in elu data set.

11.7 Conclusions

GeneCFE-ANFIS learns gene expression patterns from known genetic interactions, confirmed through biological experiments or information gathered from databases, and then GeneCFE-ANFIS can predict genetic interactions with similar nature to the known interactions. True positive rates of GeneCFE-ANFIS applied to the alpha, cdc15, cdc28 and elu data sets in [12] range from about 60% to 80%. These results are more superior to other existing approaches because their performances were usually poor when they were applied to the real MGE data. Although GeneCFE-ANFIS requires some prior knowledge to tune the inference system, but it only needs a small amount of known gene-gene interactions to yield good performances. GeneCFE-ANFIS has been tested thoroughly using the real microarray gene expression data in yeast. Moreover, the simulation results are checked against two knowledge data sets from different sources. The simulation results show that GeneCFE-ANFIS yields promising performances, and may be useful to infer gene networks.

Ackowledgements

We would like to thank Dr. Ting-Fang Wang at the Institute of Biological Chemistry, Academia Sinica, and Dr. Grace S. Shieh at the Institute of Statistical Science, Academia Sinica, for providing us with the qRT-PCR results.

References

1. Altug, S., Trussell, J., Chow, M.Y.: A mutual update training algorithm for fuzzy adaptive logic control/decision network (falcon). IEEE transactions on neural networks 10(1), 196–199 (1999)
2. Chuang, C.L., Jen, C.H., Chen, C.M., Shieh, G.S.: A pattern recognition approach to infer time-lagged genetic interactions. Bioinformatics 24(9), 1183–1190 (2008)
3. Imamura, K., Shinohara, K.: Extraction of typical and exceptional fuzzy rules from data including qualitative and quantitative attributes. In: IEEE International Conference on Fuzzy Systems, pp. 1394–1399 (1999)
4. Jang, J.S.: ANFIS: Adaptive-network-based fuzzy inference system. IEEE Transactions on System, Man and Cybernetics 23, 665–685 (1993)
5. Ji, L., Tan, K.L.: Identifying time-lagged gene clusters using gene expression data. Bioinformatics 21, 509–516 (2005)
6. Lin, F.J., Fung, R.F., Lin, H.H.: A supervisory fuzzy neural network controller for slider-crank mechanism. In: IEEE International Conference on Control Application, pp. 1710–1715 (1999)
7. Mamdani, E.H.: Application of fuzzy logic to approximate reasoning using linguistic synthesis. IEEE Trans. Computers 26, 1182–1191 (1977)
8. Ouyang, C.S., Lee, S.J.: An improved learning algorithm for rule refinement in neuro-fuzzy modeling. In: IEEE International Conference on Knowledge-based Information Engineering Systems, pp. 238–341 (1999)

9. Reis, B.Y., Butte, A.J., Kohane, I.S.: Approaching causality: discovering time-lag correlations in genetic expression data with static and dynamic relevance networks. In: Proc. of RECOMB, p. 5 (2000)
10. Schäfer, J., Strimmer, K.: An empirical bayes approach to inferring large-scale gene association networks. Bioinformatics 21, 754–764 (2005)
11. Smyth, G.K., Speed, T.: Normalization of cdna microarray data. Methods 31, 265–273 (2003)
12. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D., Futcher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast sarcharomyces cerevisiae by microarray hybridization. Mol. Biol. Cell 9, 3273–3297 (1998)
13. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. IEEE Transactions on System, Man and Cybernetics 15, 116–132 (1985)
14. Wu, A.I., Tam, P.K.S.: A simplified model of fuzzy inference system constructed by using RBF neurons. In: IEEE International Conference on Fuzzy Systems, pp. 22–25 (1999)
15. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning, I. Information Sciences 8, 199–251 (1975)
16. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning, II. Information Sciences 8, 301–357 (1975)
17. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning, III. Information Sciences 9, 43–80 (1975)

12

Scalable Dynamic Fuzzy Biomolecular Network Models for Large Scale Biology

Bahrad A. Sokhansanj[1], Suman Datta[1], and Xiaohua Hu[2]

[1] School of Biomedical Engineering, Science & Health Systems, Drexel University, 3141 Chestnut St., Philadelphia, PA 19104 USA
bahrad.sokhansanj@drexel.edu, sumandatta82@gmail.com
[2] College of Information Science and Technology, Drexel University
thu@cis.ist.drexel.edu

Summary. Fuzzy logic is an effective language for models that interpret large scale, high throughput molecular biology experiments, including genomics, proteomics, metabolomics, and inhibitor screening. Two important principles apply for biological system modeling: (1) In the post-genome era, the development of novel molecular diagnostics and therapeutics requires interpreting the complex results of high-throughput multiplexed experiments, and a framework to efficiently and rapidly design hypothesis-driven experiments. (2) Biomolecular data are typically noisy and semi-quantitative, in particular because of the typical fluorescence output of high throughput experiments. Fuzzy biomolecular network models coupled with hypothesis generation strategies address these needs. In this chapter, we describe an integrated, data-driven method for extracting system models from data and generating hypotheses for experimental design. The method is based on scalable, linear relationships between nodes of a biomolecular network, representing the expression of genes, proteins, and/or metabolites. Data from high-throughput are fuzzified using a universal normalization method. Best-fitting models are generated through an evolutionary algorithm, and disagreements between plausible hypothetical network models are used as the basis for identifying experimental designs. The result is a modeling and simulation framework that can be easily integrated with text-based and graphical biological knowledge contained within existing literature and databases.

12.1 Introduction

12.1.1 Overview of Biomolecular Network Modeling

Cells function respond to stimuli, process nutrients, repair damage, adapt to their environment through the regulated activities of DNA elements, proteins, and other molecules which constitute the structural units and catalyze chemical reactions. *Biomolecular network models* can be used to abstract the production of proteins from sequence coded in DNA, the transformation of proteins into different catalytic states, the formation and dissolution of protein complexes, and chemical interactions between proteins and small molecules. Critical functions of the cell, such as metabolism, cell cycle, stress response and repair, etc. engage biomolecular networks that include coordinated actions of hundreds to thousands of proteins.

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 235–255.
springerlink.com

Traditional molecular biological experiments were limited to studying the function of only one or small number of proteins. This knowledge base is now complemented by large data sets generated by whole genome DNA sequence information and a host of high-throughput post-genomic technologies. Examples include methods to efficiently and comprehensively measure whole genome sequences, the temporal profile of genes transcribed to proteins following a stimulus or due to a genetic perturbation (reviewed in [8]), the temporal profile of protein expression using MS [14] and protein arrays [5], the binding of proteins to DNA to regulate transcription [2], metabolite profiling through MS [11] and nuclear magnetic resonance (NMR) [12], as well as technology to rapidly generate genetic perturbations through gene silencing with RNA interference (RNAi) [25]. Understanding the biomolecular network implementing cellular function goes beyond the old dogma of one gene: one function. Only through comprehensive system understanding can we predict the impact of genetic variation in the population, design effective disease therapeutics, and evaluate the potential side-effects of therapies.

This means that mathematical modeling and computation are necessary in the emerging landscape of post-genomic biology. To understand how a complex network works together to execute a cellular function relevant for organism behavior and health demands integrating data from small-scale molecular biological experiments, high-throughput omics experiments, and phenotypic quantitative measurements and qualitative observations (at the level of cells, tissues, and the whole organism). Given the complex, heterogeneous, and multiscalar nature of this problem, it is still unclear how best to represent biological variables within a model, and given a model representation, how to identify the model based on pre-existing knowledge and experimental data. This subject has been reviewed extensively (a few examples are [9, 1, 10]). In this book chapter, we describe a fuzzy logic-based approach to both the model representation and model identification questions: (1) representing biological network interactions using fuzzy logic rule-based models, and (2) generating multiple plausible hypothetical models for a system based on a qualitative knowledge and quantitative data.

12.1.2 Motivation for Fuzzy Logic Model Representation

Boolean logic-based models of biomolecular networks have been proposed, recognizing the relationship between the logic of cellular regulation and digital circuit states with AND, OR, and IF/THEN functions [21]. However, binary rules were recognized early on in the pre-genome era to lack the dynamic resolution and range necessary to model biological function [13]. This presents a case to consider fuzzy logic as a generalization of Boolean (i.e., binary true/false) logic [40]. Rather than having an object defined absolutely within or outside a set, fuzzy logic allows for a continuum of set membership between absolutely false (defined numerically as 0.0) to absolutely true (1.0). This reflects a more biologically realistic picture of the continuum states of key variables (gene expression, protein concentration, etc.).

A variety of other approaches to state models have been implemented for gene and protein networks, including among others (references given are only examples of many): hidden Markov models (e.g., [29, 30]), Bayesian networks [28, 18], linear neural networks [7], finite state algebra [20], and probabilistic Boolean networks [31, 26]. These and other methods are based on either treating biological variables at the crudest resolution (on or off in Boolean networks, a few more levels possible for finite state models but with rapidly growing complexity) or as absolute physical quantities. Consequently, to integrate molecular biology data (generally linguistic and low-resolution), semi-quantitative data (e.g. from microarrays), and quantitative data available for biological system modeling, we focus on the fuzzy logic.

At the other end of resolution and computational complexity, differential equations and stochastic models of chemical kinetics have been proposed for biomolecular network modeling [3, 36, 39]. However, biomolecular experiments are noisy and data are semi-quantitative or qualitative: for example, pixel counts of a fluorescent spot on a membrane, chip, or microscope image. In addition, there is a cost barrier (time, technology, risk of experimental failure) associated with doing multiple experiments. Thus, in most cases, time series data for a cell or organism in response to a stress, for example are undersampled. Often, only one time point is measurement during a transition in system state, such as some arbitrary number when it is thought through other knowledge or crude preliminary experiments that system response is at its peak or has reached a quasi-steady state.

Fundamentally, biologists study systems through hypothesis generation and testing. Experiments are designed to maximize the distinction between "yes" or "no" outcomes, not to generate high-precision numbers to validate a quantitative theory. This is even true when quantitative data are obtained, for example from physiological measurements. Many chemical kinetics models are based on parameters obtained consequentially in biochemical experiments, for example [3, 34]. However, their validity as absolute physical quantities must be questioned, because the experimenters optimized *in vitro* experimental conditions to obtain statistically significant results, i.e., a maximal, reproducible difference between the results found under test and null conditions. This is a critical difference between biology, an observational science, and physics, a theoretical science. As a result, there is a formidable obstacle towards applying traditional methods used by engineers accustomed to the wealth of high-precision quantitative data on material properties to include their models of mechanical and electrical systems. Technologies to implement these methods either do not exist or are prohibitively expensive or time-consuming (and not needed to obtain novel biological insight under the hypothesis-driven or even high-throughput biology paradigms).

Fuzzy logic therefore presents an appealing bridge linking qualitative observations and knowledge to quantitative and semi-quantitative data. In particular, fuzzy logic allows for "linguistic" rule-based modeling. This allows us to apply "if/then" rules using English words to represent fuzzy states of quantitative variables [22]. Figure 12.1 shows an example of how a biochemical quantity (i.e., the

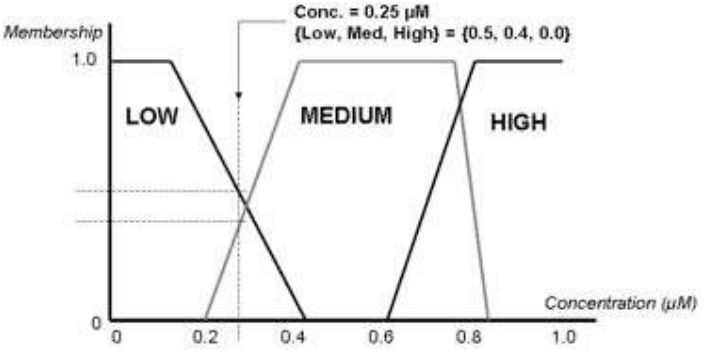


Fig. 12.1. Translation of a quantitative concentration level to a fuzzy linguistic level. In this case, the fuzzy description is a membership in the set of "low" concentration and a membership in the set of "medium". Consequently, both the state transition rules "if concentration is LOW then ..." and "if concentration is MEDIUM then ..." are evaluated.

concentration of an enzyme) can be "fuzzified" and represented by membership values in multiple fuzzy sets. For example, such fuzzy sets may be defined as "LOW", "MEDIUM", and "HIGH". Depending on the definition of fuzzy sets, a concentration such as 0.1 mM may have a membership of 0.9 in LOW, 0.1 in MEDIUM, and 0.0 in HIGH, while a concentration such as 10 mM may have a membership of 0.0 in LOW, 0.1 in MEDIUM, and 0.9 in HIGH. Then, a set of rules can be written as, for example, "if the concentration of reactant is LOW then output is MEDIUM, if input is MEDIUM then output is LOW, if input is HIGH then output is HIGH". This allows for the construction of rule-based models similar to qualitative rules found in biological literature, i.e. "if repressor gene A is expressed at a LOW level then the expression of its target gene is HIGH" and variations thereof.

12.2 General Linear Fuzzy Network Modeling

12.2.1 Developing a Scalable Method for Heterogeneous Data

Figure 12.2 shows an overview of the fuzzy modeling process. This schematic indicates the components that have to be defined for our application, including the conversion between quantitative and fuzzy values (in both directions) and the evaluation of rules based on the interaction of biomolecular network nodes. In developing a fuzzy network model representation, we are motivated by the needs for *generality*, the ability to integrate different kinds of biological data, and *scalability*, the need to avoid combinatorial explosion in modeling complex biomolecular networks. These are two of the biggest obstacles towards applying fuzzy logic to biological systems. There are many choices in developing fuzzy set definitions for a given population, such as the number, shape, range of sets.

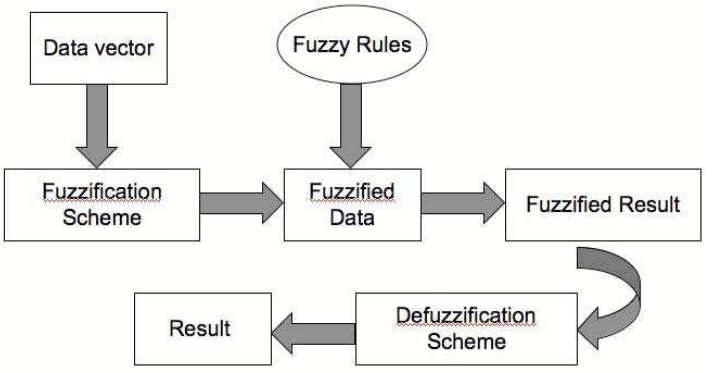


Fig. 12.2. General schematic for fuzzy logic modeling of interaction rules based on data, from data to the application of rules and the quantitative numerical interpretation of fuzzy results

Typically, the modeler develops the fuzzy set design for a given problem in collaboration with domain experts. However, to be practically applied by biologists and for rigorous use as a basis for inferring models based on experimental data, a uniform modeling framework should be capable of integrating different kinds of biological data and being applied to different regulatory systems. Once we define a fuzzy set representation scheme, the challenge becomes inference at nodes of a network model: in general, the number of possible rules scales exponentially by the number of inputs to the node and the number of fuzzy sets representing the state of each node. We have approached this problem by linearization of model inference through a method first described by Combs and Andrew (the Union Rule Configuration) [4].

12.2.2 General Fuzzification Scheme for Biological Variables

Briefly, Figure 12.3 describes the scheme used for translating semi-quantitative data to fuzzy set language. These definitions were chosen to conserve $y = x$ and $y = -x$ relationships, as well as for their conceptual simplicity and flexibility. The problem of determining appropriate fuzzy set definitions is to determine a suitable normalization of semi-quantitative values to the interval [-1,1]. Our current approach is to take the normalized arctangent of the logarithm of the expression ratios. In our previous work [6, 33], this overall scheme showed sufficient dynamic range for quantitative comparison with gene expression ratios in microarray data. In general, this is a general scheme that can be used for all kinds of ratiometric data, which are typical in biological experiments. Determining absolute quantities is in general very difficult in biology, because most measurement is based on arbitrary units (i.e., intensity fluorescence or the height of a peak in a mass spectrum). Ratiometric data are generally taken in reference to a control condition, or if data for one are not available, the mean of all measured conditions. An open question for research is the use of other methods to

deal with ratiometric data in a general fashion, but the transformation we have used can be applied the same way for proteomic data, metabolomic data, as well as genome expression, and thus can be a framework for data fusion.

12.2.3 Linear Fuzzy Relationship Functions

To reduce the problem of combinatorial explosion associated with rule-based modeling, our general modeling framework allows only a linear combination of rules at each node. That is, we follow the so-called Union Rule Configuration (URC) scheme first presented by Combs and Andrew [4]. All input rules defining the behavior of a node in the biomolecular network are thus combined by a single OR logical function. To minimize computational complexity while maintaining the quantitative resolution necessary to describe biological processes, for the OR function we employ the sum of membership functions in each fuzzy set, as described in detail in [33]. We first implemented this approach in modeling the microbial regulatory pathways of the *lac* operon (lactose metabolism) [32] and glycolysis [27]. One limitation of this approach is the inability to model nonlinear XOR interactions. This is analogous to the perceptron problem, and thus these nonlinear interactions may be resolved by hidden layers of nodes in a fashion similar to neural networks [37].

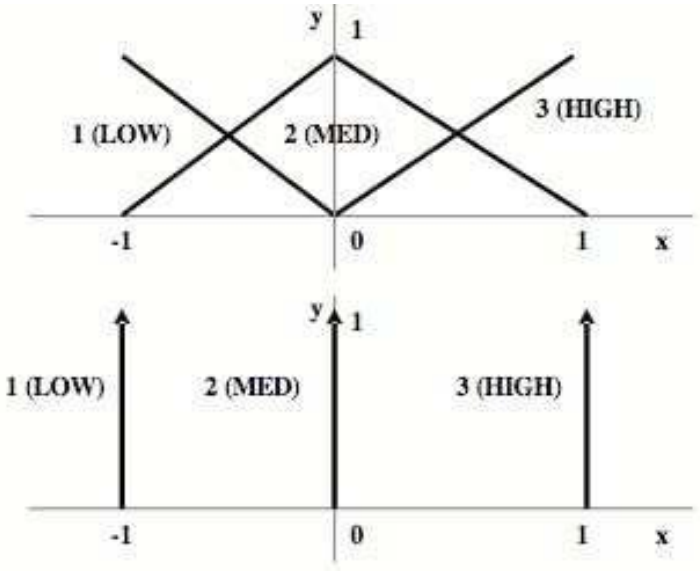


Fig. 12.3. Fuzzification (conversion from quantity to fuzzy set; top) and defuzzification (conversion back from fuzzy set to quantity; bottom) schemes. Defuzzification (applying the centroid method with point set definitions) is equivalent to dividing the difference between memberships in HIGH and LOW by the sum of memberships in all sets.

12.2.4 Model Identification: Simulating and Ranking Multiple Plausible Networks

There is no "gold standard solution" for the identifying biological systems based on data. Nor does comparing models on the basis of their agreement with previous biological knowledge provide any "proof" of success. Both the mathematical abstractions of models and the linguistic abstractions used by a biologist interpreting results in qualitative language are necessarily incomplete. The correct goal for the biological reverse engineering method should not be the "actual" model, which is ill-defined at best, but rather multiple, alternative plausible models consistent with the data. The set of plausible models may then be used to perform further simulations, complement other models, interpret biological information, and pose hypotheses for experimental study.

To compare the results of simulating a fuzzy network model with experimental data for the system, an error metric is calculated for each measurable variable from the sum of the error for each measurement of that variable (i.e., the measurement at each available time point and/or the measurement for each different experimental condition or biological stimulus being tested). The measurable variables are those gene and/or protein expression levels being measured in the available experimental data set, and they are generally restricted by practical considerations, such as which genes are spotted on a microarray, or whether proteins were measured. The other elements of the sub-network being studied act as hidden nodes within the fuzzy state model. Using the "defuzzification" scheme in Figure 12.3, fuzzy set membership values are converted to predicted levels for the measurable variables. In [33] we defined a normalized measure E of the model fit quality for each gene (node) in the gene network being simulated (perfect fit defined by $\mathbf{max}(E) = 1.0$) using the formula for predicted experimental ratio data $\{\tilde{x_i}\}$ for the node and the experimental data series $\{x_i\}$, where $\bar{x}$ is the average expression ratio over the whole experimental data series (with M data points):

$$E = \frac{\sum_{i=1}^{M}(x_i - \tilde{x_i})^2}{\sum_{i=1}^{M}(x_i - \bar{x})^2} \tag{12.1}$$

This error score was chosen to emphasize the correlation in qualitative behavior between the fit and prediction instead of the absolute numerical fit.

Thus, each hypothetical fuzzy network model is characterized by a fit quality E that represents its degree of plausibility based on available experimental data. In general, when the problem is underdetermined (i.e., more interactions than available data), there are multiple hypothetical networks that can fit the data equally well (or poorly). To illustrate this, we draw from our previous work on exhaustively searching all plausible fuzzy rules for a yeast cell cycle microarray gene expression data set. Figure 12.4, taken from [33], shows the decrease in models fitting the data a particular yeast cell cycle gene as the threshold for fit quality (E) is increased, finally terminating with a finite number of equally well-fitting hypothetical gene networks.

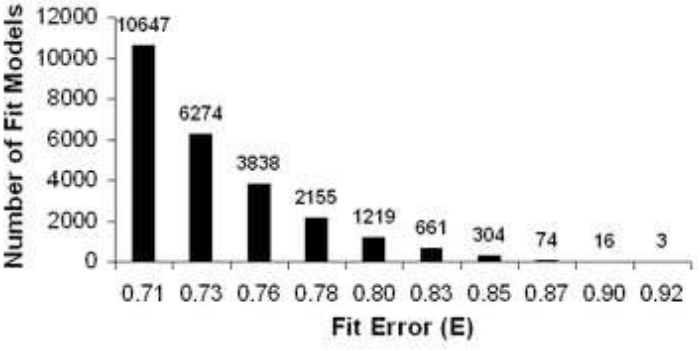


Fig. 12.4. Histogram of the number of fuzzy rules for yeast gene CLN1 from the yeast cell cycle microarray data set at different fit quality levels (indicated by E)

Subsequently, we have introduced a method to translate the E to an estimate of the probability that a rule governing a particular gene in a network model was obtained by chance [6]. This provides an assessment of the redundancy of the multiple plausible solutions to the inverse problem for biomolecular networks. As seen in our previous work doing exhaustive searches that generate all possible rule combinations, the histogram for E is roughly bell-shaped (but skewed since it is a ratio). In general, there are a few rule combinations with a low E corresponding to good fits, a large number with a moderate E that are poor fits, and a small number that represent anti-correlations, leading to a sigmoidal cumulative distribution. Based on our previous results and keeping E strictly positive, we propose to estimate the error distribution through a gamma distribution, which is defined using two parameters, a and b as,

$$y = f(x|a,b) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{-x/b}. \tag{12.2}$$

To obtain a maximum likelihood estimate of a and b for a particular variable in a given data set, we evaluate E using this equation for a *random sample of the search space* of rule combinations for that variable in that data set. As Figure 12.5 illustrates, based on our results for 20 nodes in the network, we have found that on the order of 2000 randomly selected rule combinations (of the 7^20 possibilities in this case, where 7 possible rules are allowed for each network interaction) need to be evaluated for a particular data set to obtain a stable parameter estimate. As Figure 12.6 demonstrates, we can use the shape of the probability distribution to compare the quality of data sets, in terms of to the ability to converge to a small number of plausible hypothetical networks.

Collections of hypothetical network models that remain plausible to some threshold thus define the next set of experiments that have to be performed to better understand the biomolecular network under study. What is common between the models represents a practical "ground truth" for biomolecular network knowledge, limited by the caveat of modeling limitations (only a sub-network being simulated, the limitation on number of measurements). The differences

between equally plausible hypothetical networks provide a practical basis for establishing the priorities for the next set of experiments, allowing for efficient experimental design and interpretation. Figure 12.7 shows a schematic of this process. Consequently, at least from an information/analysis perspective, biologists would not necessarily have to over-simplify the system being studied and return to the unrealistic "single gene single function" approach to biology that ignores cellular context (though the next set of experiments may have to be focused on unique players due to technical limitations but the results would still be rigorously interpreted in the broader context through modeling).

12.3 Practical Implementation

12.3.1 Challenge of Combinatorial Explosion in Larger Networks

In our earliest work, we evaluated all possible fuzzy network models for a given set of data. This allowed us to examine the qualitative relationship between fit error

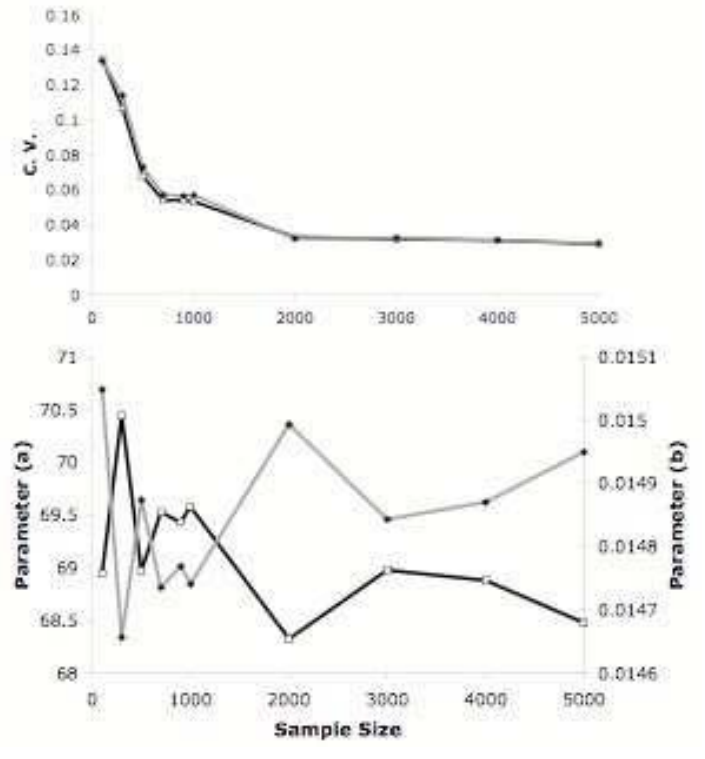


Fig. 12.5. Estimation of gamma function parameters for the error probability distribution of gene TOP2A, with the TN human cell cycle array data set. (Bottom) The mean a and b parameters (solid line with white squares and grey line with black diamonds, respectively) estimated for increasing sample sizes uniformly drawn from the space of all possible rule sets. (Top) Coefficients of variation (standard deviation divided by mean) versus sample size (based on 10 samplings).

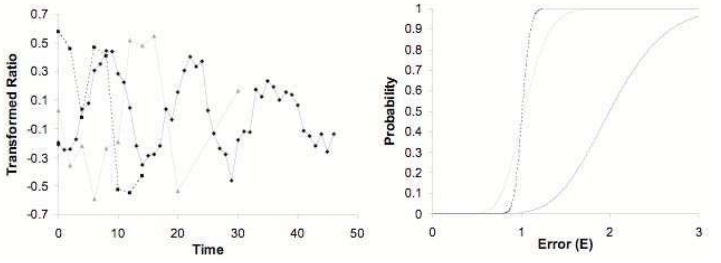


Fig. 12.6. Time series data (left) and corresponding error distribution functions for specific genes in the human cell cycle microarray data set: the solid line is data set TT3 for gene CCNB1 ($a = 22.8, b = 0.0458, a/b = 498$), the grey line is TT1 for CCNE1 ($a = 9.3, b = 0.0120, a/b = 77$), and the dashed line is Shake for CDKN3 ($a = 197, b = 0.00517, a/b = 38092$). The results are consistent with the evidence that TT3 and TT1 synchronization for cell cycle experiments leads to more robust experimental data than mechanical synchronization (i.e., the Shake data set).

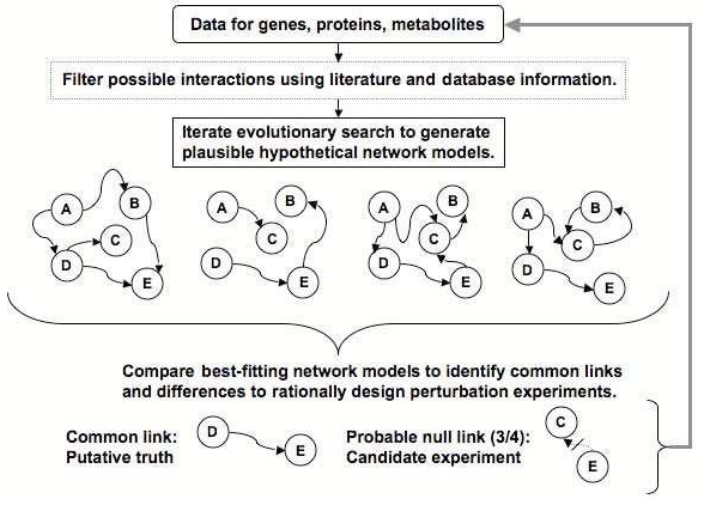


Fig. 12.7. Iterative framework for plausible biomolecular network modeling search, evaluation, and experiment generation (A, B, C, D, E are examples of biomolecules in the network). The grey arrow represents the ultimate goal of feedback between modeling predictions and generation of new experimental data.

and the number of plausible networks, i.e., developing the profile in Figure 12.4 that led to the gamma distribution described above. While employing exhaustive search, potential combinatorial explosion in the number of exhaustive hypothetical fuzzy network dynamic models generated for a biomolecular network structure has been mitigated in three ways: by using the linear fuzzy rule configuration as described above, reducing the size of the network and the setting a maximum number of possible inputs to any given node (while still allowing all other nodes

Table 12.1. Fuzzy rules allowed at a network node

Rule #	*If input is...*	*Then output is...*	*Rule #*	*If input is...*	*Then output is...*
	Low	High		Low	Low
−3	Medium	Medium	1	Medium	Medium
	High	Low		High	Medium
	Low	High		Low	Medium
−2	Medium	Medium	2	Medium	Medium
	High	Medium		High	High
	Low	Medium		Low	Low
−1	Medium	Medium	3	Medium	Medium
	High	Low		High	High

as possible inputs within that combination), this was specifically outlined in our previous work with the yeast cell cycle [15], and taking advantage of the parallel nature of the problem to run it on multiple processors. Because of the linearization of rule evaluation, each node is computed independently, so the algorithm runs for each node sequentially without depending on previous results.

In our current work, we have also considered a reduction in the number of possible rules from 27 (all 3^3 fuzzy state relationships) to just seven, including the absence of a relationship, as outlined in Table 12.1 (this is the approach used in [6]). These rules can be evaluated through simple matrix multiplications as we show here, allowing for computationally efficient rule simulation. In these examples, we show how two particular rules in Table 12.1 are evaluated to obtain the prediction for the rule on an input to the node with fuzzy set memberships 0.7 (Low), 0.3 Medium, and 0.0 (High) leading to an output with corresponding memberships in Low, Medium, and High, using the summation OR function as described above. In the case of multiple inputs, the output predictions are summed as described and the average is found (corresponding to the centroid function) to normalize set memberships to a maximum of 1.0.

$$\texttt{Rule\#}-3: \qquad \begin{bmatrix}0.7 & 0.3 & 0.0\end{bmatrix} \begin{bmatrix}0 & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{bmatrix} = \begin{bmatrix}0.0 & 0.3 & 0.7\end{bmatrix}$$

$$\texttt{Rule\#}+2: \qquad \begin{bmatrix}0.7 & 0.3 & 0.0\end{bmatrix} \begin{bmatrix}0 & 1 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}0.0 & 1.0 & 0.0\end{bmatrix}$$

In addition, as will be shown in the example problem described below, we have also begun using prior knowledge on the structure of the biomolecular network as a basis for developing dynamic models, as introduced in [17]. When used in conjunction with automated data-mining techniques to develop the prototype network model, this leads to a fully integrated fuzzy-logic based model identification and evaluation system. One approach that has not yet been implemented in our approach but we propose for future study is rule elimination through rigorous fuzzy similarity analysis as described generally in [19].

12.3.2 Evolutionary Search Algorithm for Plausible Network Models

We have recently published an evolutionary search method to accelerate the identification of plausible hypothetical network models [6]. Genetic algorithms [16] are inspired by natural selection in evolution, in which the fittest individuals in a generation pass on their genes, with probabilistic recombination and mutation. To study the feasibility of this approach, initially we have attempted one of the simplest possible evolutionary search formulations. The algorithm steps we implement are as follows, and a Matlab toolbox including scripts implementing it is freely available from the author. Figure 12.8 shows the general schematic for evolutionary optimization algorithms, which we define specifically for our application in this section.

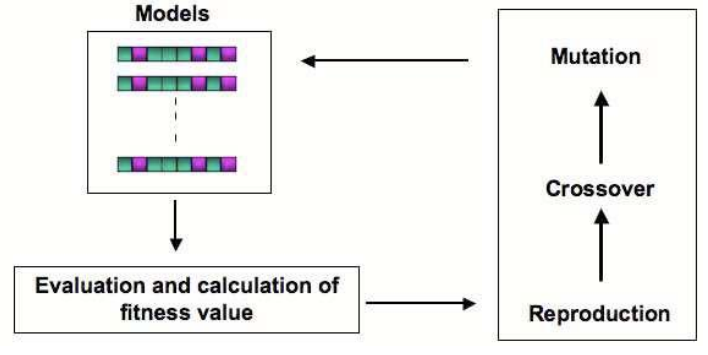


Fig. 12.8. General schematic for evolutionary algorithms, with a continuing cycle between evolution of models through genetic operators (i.e., mutation and crossover) and the selection of most fit (lowest error) for the next iteration

(1) *Generation of Initial Population.* The user can specify which rules are allowed for each possible input-output combination for the G nodes (e.g., genes) in the biomolecular network. Based on that possible space, N rule combinations are generated for the output. Each rule is a string of G randomly selected rules for each input in sequence. In this formulation, we have limited the number of possible rules to 7, roughly corresponding to different degrees of positive co-regulation, negative co-regulation, and a null rule. The null rule is introduced because the maximum input limitation is no longer in effect. In general, $N = G1$ (all other nodes can be inputs to the node being identified) but it can be constrained based on domain knowledge to reduce the search space complexity. We found based on artificial and biological data sets that population sizes of 30-50 are adequate for obtaining convergence [6]. However, it is necessary that multiple initial seeds also be considered. This is because of the large number of potentially optimal solutions, which is inherent to the problem of many possible inputs to given nodes and noisy and often sparse data sets. (2) *Evaluation of the fitness function.* The fitness of each of the N rule combinations is calculated

using the probability of fitting the rule by chance, based on gamma parameters as above. This is the most time consuming step of the process. Thus, the complexity of the algorithm scales according to the number of possible input genes and the number of data points that must be evaluated (i.e., the number of sampled points in an experimental time series). (3) *Reproduction; Crossover; Mutation.* A duplicate set of N rule combinations are generated as the Offspring Population. Within this Offspring Population, crossover and mutation are performed. All crossovers are performed before the mutations. In crossover, there is a probability p_C that each of the new rule combinations will be selected for a crossover. If one is selected, then another member of the Offspring Population is identified (from an unweighted, uniform distribution). Then, a random number of rules in the same points in the sequence of each of the two partner rule combinations (corresponding to rules for particular inputs) are selected and the rules in them are switched. (If the rules crossing over are the same, then the recombination has no effect.) In mutation, there is a probability p_M for each of the N new rule combinations to have a single mutation. If a mutation occurs, it occurs for a (uniformly distributed) random number of the inputs in the rule combination, and then another rule (circumscribed by the user-imposed constraint in Step 1 is randomly selected for each of the inputs selected for a mutation. We have identified optimal p_C and p_M as 0.6-0.7 based on artificial data sets [6]. (4) *Selection of the next generation.* The fitness function (probability of fitting the rule by chance) is calculated for each rule combination in the Offspring Population. Then, the N most fit rule combinations from the $2N$ total in the Parent and Offspring Populations are selected for the next generation, repeating from Step 3.

(5) *Termination.* The search is generally terminated when either the minimum fit error within the total population changes by less than a certain threshold, or more commonly in our implementation by attaining a certain number of generations (identified as being optimally 30-50 based on artificial data sets). This is because we view this as a means of identifying plausible network models consistent with data rather than the absolute lowest fit error which is affected by the high levels of noise in microarray and other high-throughput biological data.

Given the noise levels in biological experiments, multiple models will be consistent with the data. These models represent hypotheses, which can be constrained by prior knowledge. Modeling redundancy is a key reason why we have designed our method to be scalable for on the order of 10^2 variables (proteins and/or genes) This is because capitalization by chance will occur for larger potential solution spaces, leading to a huge number of candidate hypothetical models consistent with data sets, which are impossible to interpret. This represents a fundamental limitation that we will apply to networks at other scales as well. Thus, we have taken this limitation into account in identifying the number of search algorithm iterations, as well as the parameters for mutation and crossover operators, as detailed in [6].

12.3.3 Example: Analysis of Evolutionary Search on a Human Cell Cycle Sub-Network

Biomolecular networks are abstractions, and our identification of their components is necessarily limited. For example, mRNA microarrays measure the expression levels of genes. The co- or anti-expression of these genes suggests functional correlations, which can be termed gene regulatory networks (or "gene networks"). However, the expression of a gene relates to the rate of production of the protein it encodes. There is no information about the modification, activity, or interaction of these proteins with each other and other cellular components. Consequently, no gene network *model* will ever actually represent a *biological truth*. Within this context, to validate the use of fuzzy sets to represent semi-quantitative information, we generally test predicted "most plausible" fuzzy network models for genes against an experimental data set completely excluded from the model selection process. At each state transition, the fuzzy state values for each node (the activity or concentration of protein or expressed gene) is updated based on the values of the input nodes and the fuzzy rule base for the network being simulated.

The state transition points used in the simulation are defined by what experimental data are available and can be measured for the process being studied. In past work, for example, we simulated gene network models of the yeast (S. cerevisiae) cell cycle and compared the data to published microarray data sets with gene expression ratios for different cell cycle synchronization methods [35]. In this case, state transitions were computed at each time point measurements were made (approximately every 5 minutes over 2 hours). In the case of another study on plague bacteria microarray data, state transitions were defined at "early" (1 h), "middle" (4 h), and "late" (10 h) responses to the stimulus [27].

For the example shown here, we follow the same lines as our recently published test of the model against published human cell cycle microarray data sets obtained using different cell cycle synchronization methods [38]. The cell cycle is a particularly useful model system for studying methods of biomolecular network inference, since many genes and proteins are regulated at different phases of the cell cycle, interactions of which may be evident in the relationships between their changes over the time series. Nevertheless, there are caveats to using these kinds of data, beyond just the usual problems of noise and the lack of absolute quantification. In particular, all the cells in the sample must be synchronized so that interindividual differences in cell cycle phase do not obscure the data. We take advantage of the different synchronization methods to produce alternative data sets for training and test of inferred models. However, no synchronization method is perfect, as exemplified in Figure 12.6 by the clear differences data quality between different methods.

Here, we also employ the biomolecular identification method presented in [17]. Using several key transcription factor proteins as seeds, automated text- and data-mining methods were used to determine a protein-protein interaction networks. Then, a sub-network was found by clustering around the p300 hub protein, as shown in Figure 12.9. One of the key challenges in this approach is

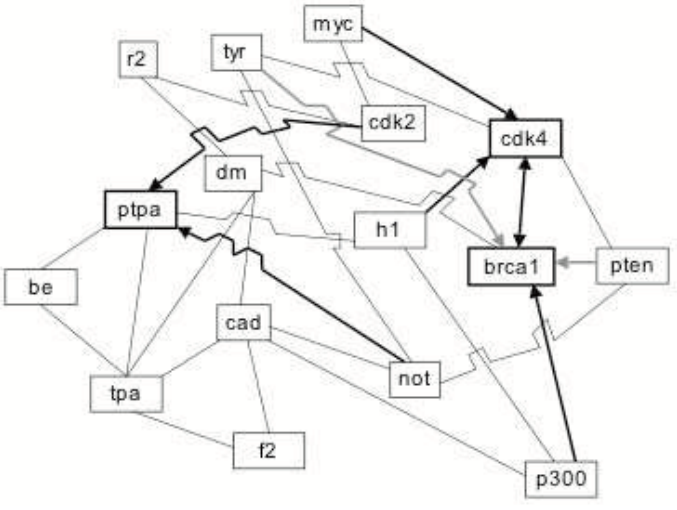


Fig. 12.9. Reduced network graph of clustered protein-protein interactions, highlighting a few of the interaction rules found in this example. Grey and black arrows indicate negative and positive feedback relationships respectively.

Table 12.2. Relationship between gene and protein names

Gene Name	Protein Name
DMPK	dm
BRCA1	brca1
H1FX	h1
PSCDBP	he
PPP2R4	ptpa
MYC	myc
NR4A2	not
F2	f2
PTEN	pt
RRM2	r2
PLAT	tpa
TYR	tyr
CAD	cad
CDK2	cdk2
CDK4	cdk4
EP300	p300

ambiguity between names given to genes and the proteins they code for. As shown in Table 12.2, which was generated by consulting with the NCBI database (`http://www.ncbi.nlm.nih.gov/sites/entrez`), there is often a non-obvious relationship between gene and protein names, with gene names being used in databases containing gene expression data. We compared network models found using "full" connectivity (all the nodes shown in Figure 12.9 can be inputs to

any other node) and the "reduced" connectivity map allowing only inputs along the edges in Figure 12.9. Then, we employed evolutionary search methods (30 generations / 30 hypothetical models evaluated in each generation) with 5 different initial seeds, selecting the best-fitting rules for each gene as an output node. The training data set is denoted ThyNoc and the test set ThyThy, corresponding to the names used by the experimenters for different cell cycle synchronization methods used to generate the data.

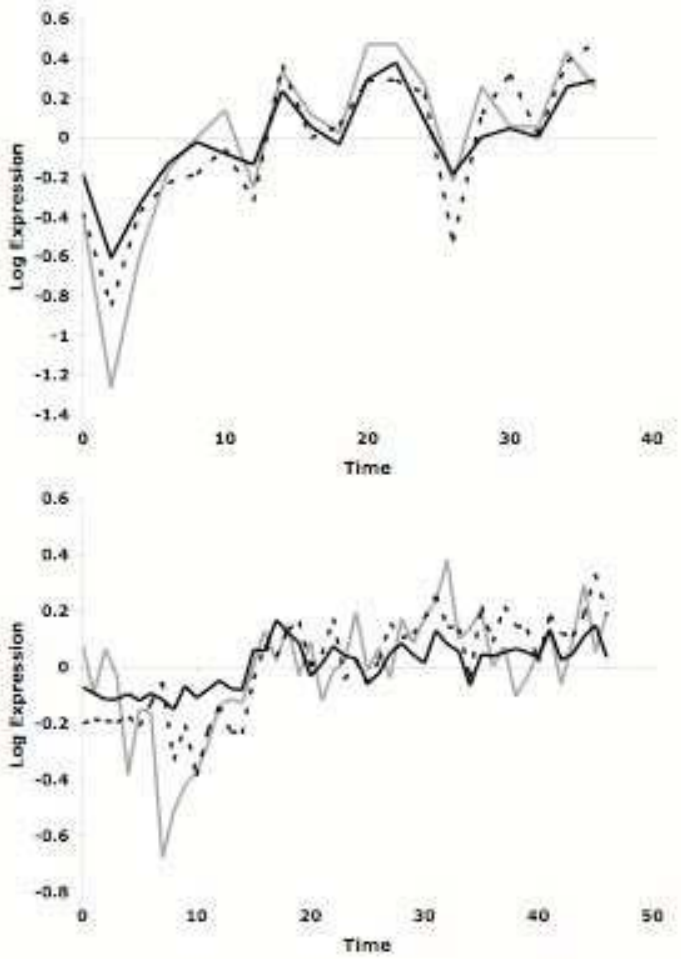


Fig. 12.10. Plots of experimental and simulated data for the CDK4 gene using the fuzzy rule set model found for the ThyNoc training data set. Shown here are (top) comparing the predicted curves for the full connectivity matrix (solid black), reduced network-based connectivity matrix (dashed black), and ThyNoc experimental data (solid grey), and (bottom) the same predicted rules (obtained by training on the ThyNoc data set) tested on the ThyThy experimental data set.

Table 12.3. Fuzzy rule models found by evolutionary search: output genes are in rows, and input genes are in columns following the same sequence. The first row for each gene is the best-fitting rule found for the assumption of full connectivity, and the subsequent row is the best-fitting rule assuming the reduced connectivity model shown in Figure 12.7.

DMTF	0	-2	0	0	0	0	0	0	3	-2	0	-1	-3	3	3	-1
DMTF	0	-1	1	0	0	0	0	0	0	-2	0	0	-2	0	0	0
BRCA1	0	0	0	0	0	0	0	-2	-3	0	0	-3	0	2	1	1
BRCA1	0	0	0	0	0	0	0	0	-3	0	0	-3	0	0	3	3
H1FX	0	0	0	0	0	3	0	0	0	3	3	-3	1	0	3	0
H1FX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0
HE	0	0	2	0	-3	-3	0	1	0	-2	3	0	-3	0	0	-3
HE	0	0	0	0	-3	0	0	0	0	0	2	0	0	0	0	0
PPP2R4	0	3	1	-3	0	3	1	0	1	3	0	-3	3	3	3	3
PPP2R4	0	0	0	0	0	0	1	0	0	0	0	0	0	3	0	0
MYC	0	-1	0	-2	0	0	2	-3	0	0	-3	0	3	0	3	3
MYC	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0
NR4A2	-2	1	-3	0	0	3	0	0	-3	3	1	0	2	-3	0	3
NR4A2	0	0	0	0	0	0	0	-1	0	0	0	2	0	0	0	3
F2	0	-3	-2	1	0	-3	0	0	2	0	0	1	-1	0	-3	-2
F2	0	0	0	0	0	0	0	0	0	0	0	-2	0	-3	0	0	0
PTEN	3	-3	0	0	0	0	-1	3	0	3	-3	0	0	1	-1	0
PTEN	0	0	0	0	0	0	0	0	0	0	0	0	0	1	-3	0
RRM2	-1	0	3	-3	3	-2	0	0	3	0	0	0	2	3	0	0
RRM2	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
PLAT	0	0	3	3	0	-3	0	0	-2	0	0	0	0	0	3	0
PLAT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYR	0	-3	-2	3	-3	3	3	3	0	0	0	0	-1	-2	-3	-3
TYR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-2	0
CAD	0	0	3	-2	0	3	2	-3	-3	0	1	0	0	0	3	3
CAD	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
CDK2	3	1	-1	-3	3	-2	0	0	2	3	-1	-3	2	0	3	0
CDK2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CDK4	0	3	3	0	1	3	0	0	-2	0	3	-3	3	0	0	0
CDK4	0	3	3	0	0	3	0	0	0	0	0	0	0	0	0	0
EP300	0	3	-3	-3	3	0	1	-3	-1	1	0	-3	0	-2	0	0
EP300	0	3	-1	0	0	0	3	0	0	0	0	0	2	0	0	0

Table 12.3 shows the rules that were found, comparing the plausible models found using the full connectivity and reduced connectivity map. In keeping with the linearization used to enhance scalability, the rules here are connected by OR (summation) relationships. Notably, there is at least directional agreement between common interactions that were identified in both cases. Also, where there is a connection suggested by the full connectivity assumption and not found by the prior data-mining, a hypothesis can be generated that the connectivity map in Figure 12.9 is incomplete. The partial overlaying of the rules from Table 12.3

Table 12.4. Evaluation of errors of models found by evolutionary search

		ThyNoc			**ThyThy**	
Gene	Full	Reduced	E for $P = 10^-5$	Full	Reduced	E for $P = 10^-5$
DMTF	0.6055	0.7106	0.6156	0.9404	1.4629	0.6219
BRCA1	0.4514	0.3792	0.4876	1.3448	1.8728	0.3941
H1FX	0.5708	0.3319	0.67	1.0118	0.6115	0.7748
HE	0.4046	0.5031	0.5211	1.2915	2.0834	0.6954
PPP2R4	0.3168	0.5053	0.3832	0.6844	0.8911	0.4925
MYC	0.6144	0.4267	0.6553	0.9905	1.0131	0.5799
NR4A2	0.5626	0.5494	0.4752	0.7465	1.0467	0.4274
F2	0.3163	0.6274	0.3394	1.0768	1.3202	0.7152
PTEN	0.6476	0.801	0.6969	1.0773	1.0849	0.6846
RRM2	0.6919	0.7445	0.688	0.9529	0.7462	0.7669
PLAT	0.4497	1.0203	0.6543	1.3132	1.0014	0.7546
TYR	0.3576	0.8752	0.4258	1.2229	2.2915	0.5403
CAD	0.3703	0.9529	0.4274	0.8565	1.4029	0.4058
CDK2	0.4407	1.031	0.4746	0.7312	1	0.5474
CDK4	0.1934	0.184	0.3636	0.6341	0.6282	0.5158
EP300	0.4707	0.5337	0.4607	1.1111	1.586	0.735

on the map in Figure 12.9 shows the potential for fuzzy modeling to be used to provide dynamic interaction information from an experimental data set on top of the static regulatory map inferred from biological knowledge. Table 12.4 shows the error of the fits, (ThyNoc) and test (ThyThy) data sets, each of which represent different synchronization methods with different numbers of data points, and show the general consistency between error of fitting on training and test data sets. To make it easier to interpret error values, we also indicate for each data set the error score corresponding to rules that occur with a frequency of 10^{-5} being found in the search space, as determined by Eqn. (12.2). The fit for the gene CDK2 on the training and test sets (for both the full connectivity and reduced connectivity assumptions) is shown by time series plots in Figure 12.10.

12.4 Conclusions

The example we show here provides a specific example of general challenges faced by fuzzy logic and other efforts to infer knowledge from large-scale experiments. In particular, the search space for models that are consistent with experimental data contains many similarly optimal solutions (as shown by the generally low P values calculated by Eqn. (12.2) and shown in Table 12.4). This provides additional concrete evidence of the fallacy of finding an optimal reverse-engineered biomolecular network model based on data analysis alone. Even when prior biological knowledge is added (as in the kind of static interaction map shown in Figure 12.7), there are still many plausible network hypotheses, however. Thus, there is a need for high-throughput experiments guided by the analysis of modeling results, as suggested by the schematic in Figure 12.6. One of the profound

advantages of the method described here is the ability to explicitly identify plausible hypotheses that can be ranked for experimental priority.

Another advantage suggested by the integration with text- and data-mining algorithms is the ability of fuzzy logic models to be translated to the linguistic and graphical ways of understanding systems familiar to biologists. There is a clear tie between the natural language processing required to interface with hypothesis-based biological experiments and the linguistic models of dynamic interactions measured in high throughput experiments. This is becoming particular significant as large-scale data sets are being gathered to separate diseases by molecular signatures. There are significant technical issues with these molecular signatures [23], requiring us to employ biological knowledge to understand what the most important elements of a signature are based on function, and how multiple signatures may overlap with each other because they fall in the same regulatory circuit. Another crucial application will be the emerging field of "polypharmacology", which is motivated by the finite limit of "druggable" protein targets within cells: determining multiple targets for drugs, and combinations of drugs targeting combinations of targets [24].

References

1. Arita, M., Robert, M., Tomita, M.: All systems go: launching cell simulation fueled by integrated experimental biology data. Curr. Opin. Biotech. 16, 344–349 (2005)
2. McCutchen-Maloney, S.L., Forde, C.E.: Characterization of transcription factors by mass spectrometry and the role of seldi-ms. Mass. Spectrom. Rev. 21, 419–439 (2002)
3. Chen, K.C., Csikasz-Nagy, A., Gyorffy, B., Val, J., Novak, B., Tyson, J.: Kinetic analysis of a molecular model of the budding yeast cell cycle. Mol. Biol. Cell. 13, 52–70 (2000)
4. Combs, W.E., Andrews, J.E.: Combinatorial rule explosion eliminated by a fuzzy rule configuration. IEEE Trans. Fuzzy Syst. 6, 1–11 (1998)
5. Cutler, P.L.: Protein arrays: the current state-of-the-art. Proteomics 3, 3–18 (2003)
6. Datta, S., Sokhansanj, B.A.: Accelerated search for biomolecular network models to interpret high-throughput experimental data. BMC Bioinformatics 8, 258 (2007)
7. D'Haeseleer, P., Wen, X., Fuhrman, S., Somogyi, R.: Linear modeling of mrna expression levels during cns development and injury. In: Pac. Symp. Biocomp (PSB 1999), vol. 2, pp. 41–52 (1999)
8. Fitch, J.P., Sokhansanj, B.: Genomic engineering: moving beyond dna sequence to function. Proc. IEEE 88, 1949–1971 (2000)
9. Friedman, N.: Inferring cellular networks using probabilistic graphical models. Science 303(5659), 799–805 (2004)
10. Gianchandani, E.P., Brautigan, D.L., Papin, J.A.: System analyses characterize integrated functions of biochemical networks. Trends Biochem. Sci. 31, 284–291 (2006)
11. Gipson, G.T., Tatsuoka, K.S., Sokhansanj, B.A., Ball, R.J., Connor, S.C.: Assignment of ms-based metabolomic datasets via compound interaction pair mapping. Metabolomics 4, 94–103 (2008)
12. Gipson, G.T., Tatsuoka, K.S., Sweatman, B.C., Connor, S.C.: Weighted least-squares deconvolution method for discovery of group differences between complex biofluid 1h nmr spectra. J. Magn. Reson. 183, 269–277 (2006)

13. Glass, L., Kauffman, S.A.: The logical analysis of continuous, nonlinear biochemical control networks. J. Theor. Biol. 39, 103–129 (1973)
14. Griffin, T.J., Gygi, S.P., Ideker, T., Rist, B., Eng, J., Hood, L., Aebersold, R.: Complementary profiling of gene expression at the transcriptome and proteome levels in saccharomyces cerevisiae. Mol. Cell. Proteomics 1, 323–333 (2002)
15. Grigoriev, A.: On the number of protein-protein interactions in the yeast proteome. Nucleic Acids Res. 31, 4157–4161 (2003)
16. Holland, J.H.: Adaptation in natural and artificial systems. University of Michigan Press (1975)
17. Hu, X., Wu, D.D.: Data mining and predictive modeling of biomolecular network from biomedical literature and databases. IEEE/ACM Trans. Comput. Biol. Bioinform. 4, 251–263 (2007)
18. Husmeier, D.: Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic bayesian networks. Bioinformatics 19, 2271–2282 (2003)
19. Jin, Y.: Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement. IEEE Trans. Fuzzy Syst. 8, 212–221 (2000)
20. Laubenbacher, R., Stigler, B.: A computational algebra approach to the reverse engineering of gene regulatory networks. J. Theor. Biol. 229, 523–537 (2004)
21. Liang, S., Fuhrman, S., Somogyi, R.: REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. In: Pac. Symp. Biocomp (PSB 2000), vol. 3, pp. 18–29 (2000)
22. Mendel, J.M.: Fuzzy logic systems for engineering: a tutorial. Proc. IEEE 83, 345–377 (1995)
23. Michiels, S., Koscielny, S., Hill, C.: Interpretation of microarray data in cancer. Br. J. Cancer. 96, 1155–1158 (2007)
24. Overington, J.P., Al-Lazikani, B., Hopkins, A.L.: How many drug targets are there? Nat. Rev. Drug. Disc. 5, 993–996 (2006)
25. Paddison, P.J., Silva, J.M., Conklin, D.S., Schlabach, M., Li, M., Aruleba, S., Balija, V., O'Shaughnessy, A., Gnoj, L., Scobie, K., Chang, K., Westbrook, T., Cleary, M., Sachidanandam, R., McCombie, W.R., Elledge, S.J., Hannon, G.J.: A resource for large-scale rna-interference-based screens in mammals. Nature 428, 427–431 (2004)
26. Perkins, T.J., Hallett, M., Glass, L.: Inferring models of gene expression dynamics. J. Theor. Biol. 230, 289–299 (2004)
27. Quong, A.A., Kercher, J.R., McCready, P.M., Quong, J.N., Sokhansanj, B.A., Fitch, J.P.: An indexed modeling and experimental strategy for biosignatures of pathogen and host. J. Franklin. Inst. 341, 157–174 (2004)
28. Rangel, C., Angus, J., Ghahramani, Z., Lioumi, M., Sotheran, E., Gaiba, A.: Modeling t-cell activation using gene expression profiling and state-space models. Bioinformatics 20, 1361–1372 (2004)
29. Rosales, R.A., Fill, M., Escobar, A.L.: Calcium regulation of single ryanodine receptor channel gating analyzed using hmm/mcmc statistical methods. J. Gen. Physiol. 121, 533–553 (2004)
30. Schliep, A., Schonhuth, A., Steinhoff, C.: Using hidden markov models to analyze gene expression time course data. Bioinformatics 19, i255–i263 (2003)
31. Shmulevich, I., Dougherty, E.R., Kim, S., Zhang, W.: Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks. Bioinformatics 18, 261–274 (2002)
32. Sokhansanj, B.A., Fitch, J.P.: URC fuzzy modeling and simulation of gene regulation. In: Conf. Proc. IEEE Eng. Med. Biol. Soc., vol. 3, pp. 2918–2921 (2001)

33. Sokhansanj, B.A., Fitch, J.P., Quong, J.N., Quong, A.A.: Linear fuzzy gene network models obtained from microarray data by exhaustive search. BMC Bioinformatics 5, 108 (2004)
34. Sokhansanj, B.A., Rodrigue, G.R., Fitch, J.P., Wilson III, D.M.: A quantitative model of human dna base excision repair. i. mechanistic insights. Nucleic Acids Res 30, 1817–1825 (2002)
35. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D., Fucher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. Mol. Biol. Cell. 9, 3273–3297 (1998)
36. Tegner, J., Yeung, M.K.S., Hasty, J., Collins, J.J.: Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. Proc. Nat. Acad. Sci. USA 99, 6163–6168 (2000)
37. Weinschenk, J.J., Marks, R.J.I., Combs, W.E.: Layered urc fuzzy systems: a novel link between fuzzy systems and neural networks. In: Proc. 2003 Intl. Joint Conf. Neural Net, pp. 2995–3000 (2003)
38. Whitfield, M.L., Sherlock, G., Saldanha, A.J., Murray, J.I., Ball, C.A., Alexander, K.E., Matese, J.C., Perou, C.M., Hurt, M.M., Brown, P.O., Botstein, D.: Identification of genes periodically expressed in the human cell cycle and their expression in tumors. Mol. Biol. Cell 13, 1977–2000 (2002)
39. Yeung, M.K.S., Tegner, J., Collins, J.J.: Reverse engineering gene networks using signular value decomposition and robust recognition. Proc. Nat. Acad. Sci. USA 100, 5944–5949 (2002)
40. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–352 (1965)

13

Fuzzy C-Means Techniques for Medical Image Segmentation

Huiyu Zhou[1], Gerald Schaefer[2], and Chunmei Shi[3]

[1] School of Engineering and Design, Brunel University, Uxbridge, U.K.
Huiyu.Zhou@brunel.ac.uk
[2] School of Engineering and Applied Science, Aston University, Birmingham, U.K.
G.Schaefer@aston.ac.uk
[3] Peoples Hospital of Guangxi, Nanning, China

Summary. Segmentation is an important step in many medical imaging applications and a variety of image segmentation techniques exist. One group of segmentation algorithms is based on clustering concepts. In this chapter we provide an overview of several fuzzy c-means based clustering approaches and their application to medical imaging. In particular we evaluate the conventional hard c-means and fuzzy c-means (FCM) approches as well as three computationally more efficient derivatives of fuzzy c-means: fast FCM with random sampling, fast generalised FCM, and a new anisotropic mean shift based FCM.

13.1 Introduction

Clinical applications typically require image segmentation so that e.g. different anatomical parts or biological tissues can be easily identified. Unfortunately, segmentation has proven to be a very hard problem due to the diversity in modalities and image characteristics as well as image noise and image artefacts. For example, magnetic resonance imaging may suffer from the irregularities of the magnetic fields leading to intensity inhomogenities while speckle noise in ultrasonic images can induce some image sections to become disconnected. Development of an 'optimal' segmentation methodology is hence highly sought after and has attracted much attention in the research community.

Image segmentation can be defined as the grouping of similar pixels in a parametric space, where they are associated with each other in the same or different images. Classical image segmentation methodologies include thresholding, edge detection, and region detection [12]. Thresholding methods are relatively simple but lack sensitivity and specificity for accurate segmentation in the presence of different objects with similar intensities or colours. Edge-based methods, quite similar to the contour detection, are fast but sensitive to noise in the background and fail to link together broken contours. While region detection is superior to thresholding and edge-based methods in terms of stability and consistency, nevertheless, these approaches need further modifications in order to effectively handle e.g. image occlusions which commonly exist in real scenarios. Better segmentation is achieved by connectivity-preserving relaxation methods, also referred to as the

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 257–271.
springerlink.com

active contour models [15], which start from an initial contour shape, followed by applying shrink or expansion operations according to a defined object function. However, here the convergence of the computation is affected by local minima of the function which hence might lead to incorrect segmentation.

Image segmentation can also be approached as a clustering problem. Conventional hard c-means (HCM) and fuzzy c-means (FCM) algorithms are two clustering-based [13] segmentation techniques. In contrast to HCM, FCM allows us to reduce the uncertainty of pixels belonging to one class and therefore in general provides improved segmentation. In addition, multiple classes with varying degrees of membership can be continuously updated. In recent years, numerous efforts of c-means/k-means segmentation methods with faster computation and more flexible capabilities than the classical techniques, have emerged.

In this chapter, we provide an overview of the more popular c-means based segmentation techniques, namely HCM [18], conventional FCM [2], fast FCM with random sampling [6], fast generalized FCM [20], and also present a new anisotropic mean shift based fuzzy c-means algorithm [23]. The proposed clustering method incorporates a mean field term within the standard fuzzy C-means objective function. Since mean shift can quickly and reliably find cluster centres, the entire strategy is capable of optimally segmenting clusters within the image.

13.2 Hard c-Means

Hard c-means clustering (HCM) is one of the most widely used unsupervised algorithms used to solve the well known clustering problem [18]. The algorithm exploits a simple but effective way to handle the classification problem of a given data set given a preset number of C clusters. The general idea is to identify C centroids, one for each cluster. The locations of these centroids are critical as they directly determine the clustering results.

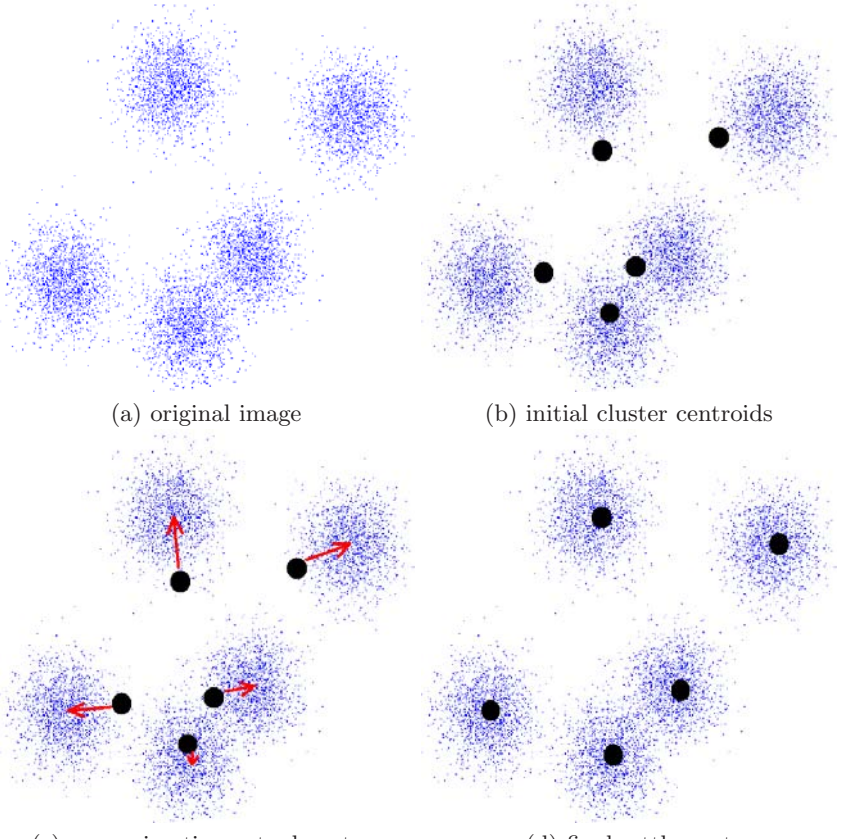
C-means clustering sets out to minimise an objective function in squared error form

$$E = \sum_{j=1}^{C} \sum_{i=1}^{N} ||x_i^{(j)} - c_j||^2 \tag{13.1}$$

where $||x_i^{(j)} - c_j||^2$ is the distance between one of the N data samples x_i and its closest cluster centroid c_j. In order to minimise E the following procedure is followed:

Step 1: Initialise C cluster centroids.
Step 2: Associate each sample (i.e. pixel) with the centroid closest to it.
Step 3: Re-compute the cluster centroids based on the mapped samples.
Step 4: Repeat steps 2 to 3 until convergence, i.e., until the locations of the centroids do not change.

This process is illustrated in Figure 13.1 based on an example with five clusters.



(a) original image (b) initial cluster centroids

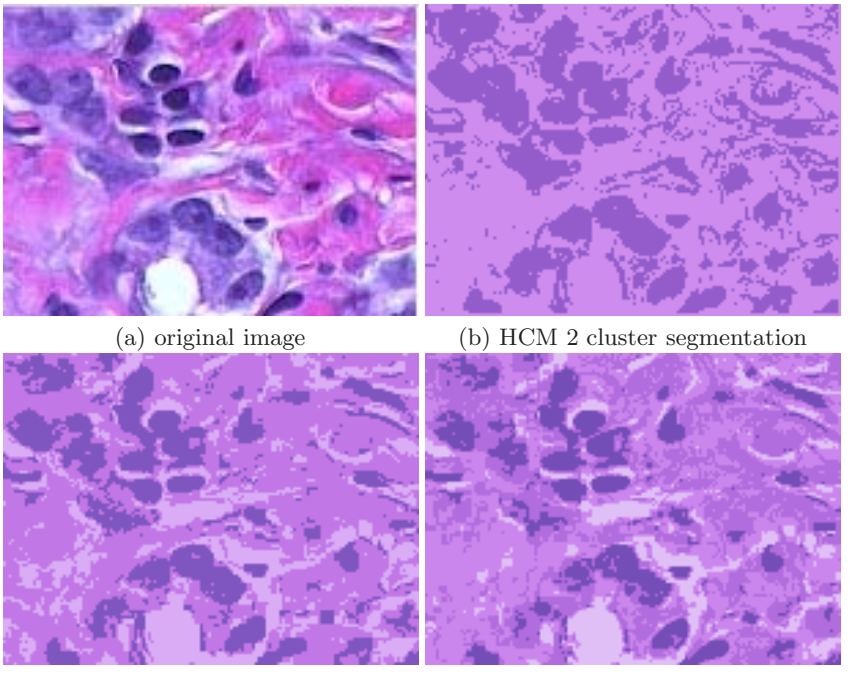
(c) approximating actual centres (d) final settlement

Fig. 13.1. Illustration of c-means clustering algorithm. Dots represent sample and discs the cluster centres.

An example of c-means clustering on medical images is given in Figure 13.2(a) which shows an image of tissue stained with hemotoxylin and erosin. This image was used to help pathologists distinguish several tissue types. Figure 13.2(b)-(d) show the results of c-means clustering with different numbers of clusters (2, 3, and 4).

It should be noted that despite guaranteed convergence, HCM typically does not find the optimal solution. This is due to the fact that the algorithm only converges towards the local minimum and hence the algorithm is significantly affected by the starting configuration of cluster centres.

Although c-means represents a simple and well established method for clustering and segmentation, experience shows that it is necessary to further improve this methodology. For example, Dhillon *et al.* [10] noticed that cosine similarity based k-means did not work effectively when applied to document collections. A strategy of re-assigning samples and immediately re-computing centroids can



(a) original image (b) HCM 2 cluster segmentation

(c) HCM 3 cluster segmentation (d) HCM 4 cluster segmentation

Fig. 13.2. Example of stained tissue image and segmentations based on c-means clustering with 2, 3, and 4 clusters. Original image is courtesy of John Hopkins University.

work better. To address the cluster initialisation problem, Bradley and Fayyad [3] proposed to randomly sample a number of data points for c-means. Each of these constructed systems was then evaluated, where the best system was used to initiate HCM on the entire data set. Zhang [22] presented a rectified optimisation process using soft assignment of image points to different clusters with proper weights rather than directly moving them from one cluster to another. In terms of distance measurements, the Mahalanobis distance was used to handle hyper-ellipsoidal clusters in [19].

13.3 Fuzzy c-Means

Fuzzy c-means (FCM) is based on the same idea of finding cluster centres by iteratively adjusting their positions and evaluation of an objective function as HCM, yet it allows more flexibility by introducing the possibility of partial memberships to clusters. The error function from Equation (13.1) is thus extended to

$$E = \sum_{j=1}^{C} \sum_{i=1}^{N} \mu_{ij}^{k} ||x_i^{(j)} - c_j||^2 \tag{13.2}$$

where μ_{ij} is the fuzzy membership of sample (or pixel) x_i and the cluster identified by its centre c_j, and k is a constant that defines the fuzziness of the resulting partitions.

E can reach the global minimum when pixels nearby the centroid of corresponding clusters are assigned higher membership values, while lower membership values are assigned to pixels distant from the centroid [7]. In here, the membership is proportional to the probability that a pixel belongs to a specific cluster where the probability is only dependent on the distance between the image pixel and each independent cluster centre. The membership functions and the cluster centres are updated by

$$\mu_{ij} = \frac{1}{\sum_{m=1}^{C} \left(\frac{||x_j - c_i||}{||x_j - c_m||)^{2/(k-1)}} \right)} \tag{13.3}$$

and

$$c_i = \frac{\sum_{j=1}^{N} \mu_{ij}^k x_j}{\sum_{j=1}^{N} \mu_{ij}^k} \tag{13.4}$$

The steps involved in fuzzy c-means image segmentation are [2]:

Step 1: Initialise the cluster centres c_i and let $t = 0$.
Step 2: Initialise the fuzzy partition memberships functions μ_{ij} according to Equation (13.3).
Step 3: Let $t = t + 1$ and compute new cluster centres c_i using Equation (13.4).
Step 4: Repeat Steps 2 to 3 until convergence.

Again, an initial setting for each cluster centre is required and FCM also converges to a local minimum. The efficiency of FCM has been investigated in [14]. To effectively address the inefficiency of the algorithm several variants of the fuzzy c-means algorithm have been introduced in the literature.

13.4 Fast FCM with Random Sampling (RSFCM)

To combat the computational complexity of FCM, Cheng *et al.* [6] proposed a multistage random sampling strategy. This method has a lower number of feature vectors and also requires fewer iterations to converge. The basic idea is to randomly sample and obtain a small subset of the dataset in order to approximate the cluster centres of the full dataset. This approximation is then used to reduce the number of iterations. Random sampling FCM (RSFCM) consists of two phases. First, a multistage iterative process of a modified FCM is performed. Phase 2 is then a standard FCM with the cluster centres approximated by the final cluster centres from Phase 1.

Phase 1:

Let $X_{\Delta\%}$ be a subset whose number of subsamples is $\Delta\%$ of the N samples contained in the full dataset X and denote the number of stages as n. ϵ_1 and ϵ_2 are parameters used as stopping criteria. After the following steps the dataset (denoted as $X_{(n_s * \Delta\%)}$) will include $N * \Delta\%$ samples:

Step 1: Select $X_{(\Delta\%)}$ from the set of the original feature vectors matrix ($z = 1$).
Step 2: Initialise the fuzzy memberships functions μ using Equation (13.3) with $X_{(z*\Delta\%)}$.
Step 3: Compute the stopping condition $\epsilon = \epsilon_1\text{-z}*((\epsilon_1\text{-}\epsilon_2)/n_s)$ and let $t = 0$
Step 4: Set $t = t + 1$
Step 5: Compute the cluster centres $c_{(z*\Delta\%)}$ using Equation (13.4).
Step 6: Compute $\mu_{(z*\Delta\%)}$ using Equation (13.3).
Step 7: If $||\mu^j_{(z*\Delta\%)} - \mu^{j-1}_{(z*\Delta\%)}|| \geq \epsilon$, then go to Step 4.
Step 8: If $z \leq n_s$ then select another $X_{(\Delta\%)}$ and merge it with the current $X_{(z*\Delta\%)}$ and set $z = z + 1$, otherwise move to Phase 2 of the algorithm.

Phase 2:

Step 1: Initialise μ_{ij} using the results from Phase 1, i.e. $c_{(n_s*\Delta\%)}$ with Equation (13.4) for the full data set
Step 2: Go to Steps 3 of the conventional FCM algorithm and iterate the algorithm until stopping criterion ϵ_2 is met.

Evidence has shown that this improved FCM is able to reduce the computation requested in the classical FCM method. Other variants of this multistage random sampling FCM framework have also been developed and can be found e.g. in [11] and [16].

13.5 Fast Generalized FCM Scheme (EnFCM)

Ahmed *et al.* [1] introduced an alternative to the classical FCM by adding a term that enables the labelling of a pixel to be associated with its neighbourhood. As a regulator, the neighbourhood term can change the solution towards piecewise homogeneous labelling. As a further extension of this work, Szilágyi *et al.* [20] proposed their EnFCM algorithm to speed up the segmentation process for black-and-white images. In order to reduce the computational complexity, a linearity-weighted sum image g is formed from the original image, and the local neighbour average image evaluated as

$$g_m = \frac{1}{1+\alpha}\left(x_m + \frac{\alpha}{N_R}\sum_{j\in N_r} x_j\right) \tag{13.5}$$

where g_m denotes the gray value of the m-th pixel of the image g, x_j represents the neighbours of x_m, N_R is the cardinality of a cluster, and N_r represents the set of neighbours falling into a window around x_m.

The objective function used for segmenting image g is defined as

$$J = \sum_{i=1}^{C}\sum_{i=1}^{q_c}\gamma_l\mu_{ij}^m(g_l - c_i)^2 \tag{13.6}$$

where q_c denotes the number of the gray levels in the image, and γ_l is the number of the pixels having an intensity equal to l with $l = 1, 2, \ldots, q_c$. Thus, $\sum_{l=1}^{q_c} \gamma_l = N$ under the constraint that $\sum_{i=1}^{C} \mu_{ij} = 1$ for any l.

Finally, we can obtain the following expressions for membership functions and cluster centres [4]:.

$$\mu_{il} = \frac{(g_l - c_i)^{-2/m-1}}{\sum_{j=1}^{C}(g_l - c_j)^{-2/m-1}} \tag{13.7}$$

and

$$s_i = \frac{\sum_{l=1}^{q_c} \gamma_l \mu_{il}^m g_l}{\sum_{l=1}^{q_c} \gamma_l \mu_{il}^m} \tag{13.8}$$

EnFCM considers a number of pixels with similar intensities as a weight. Thus, this process may accelerate the convergence of searching for global similarity. On the other hand, to avoid image blur during the segmentation, which may lead to inaccurate clustering, Cai *et al.* [4] utilised a measure S_{ij}, which incorporates the local spatial relationship S_{ij}^s and the local gray-level relationship S_{ij}^g, and is defined as

$$S_{ij} = \begin{cases} S_{ij}^s \times S_{ij}^g, & j \neq i \\ 0, & j = i \end{cases} \tag{13.9}$$

with

$$S_{ij}^s = \exp\left(\frac{-\max(|p_{cj} - p_{ci}|, |q_{cj} - q_{ci}|)}{\lambda_s}\right) \tag{13.10}$$

and

$$S_{ij}^g = \exp\left(\frac{-||x_i - x_j||^2}{\lambda_g \times \sigma_g^2}\right) \tag{13.11}$$

where (p_{ci}, q_{ci}) describe the co-ordinates of the i-th pixel, σ_g is a global scale factor of the spread of S_{ij}^s, and λ_s and λ_g represent scaling factors. S_{ij} replaces α in Eq. (13.5).

Hence, the newly generated image g is updated as

$$g_i = \frac{\sum_{j \in N_i} S_{ij} x_j}{S_{ij}} \tag{13.12}$$

and is restricted to [0, 255] due to the denominator.

Given a pre-defined number of clusters C and a threshold value $\epsilon > 0$, the fast generalised FCM algorithm proceeds in the following steps:

Step 1: Initialise the clusters c_j.
Step 2: Compute the local similarity measures S_{ij} using Equation (13.9) for all neighbours and windows over the image.
Step 3: Compute the linearly-weighted summed image g using Equation (13.12).
Step 4: Update the membership partitions using Equation (13.7).
Step 5: Update the cluster centres c_i using Equation (13.8).
Step 6: If $\sum_{i=1}^{C} ||c_{i(old)} - c_{i(new}||^2 > \epsilon$ go to Step 4.

Similar efforts to improve the computational efficiency and robustness have also been reported in [17] and [5].

13.6 Anisotropic Mean Shift Based FCM (AMSFCM)

In the following we will present a new efficient approach to fuzzy c-means clustering that utilises an anisotropic mean shift algorithm coupled with fuzzy clustering [23].

Mean shift based techniques have been shown to be capable of estimating the local density gradients of similar pixels. These gradient estimates are iteratively performed so that all pixels can find similar pixels in the same image [8, 9]. A standard mean shift approach method uses radially symmetric kernels. Unfortunately, the temporal coherence will be reduced in the presence of irregular structures and noise in the image. This reduced coherence may not be properly detected by radially symmetric kernels and thus, an improved mean shift approach, namely anisotropic kernel mean shift [21], provides better performance.

In mean shift algorithms the image clusters are iteratively moved along the gradient of the density function before they become stationary. Those points gathering in an outlined area are treated as the members of the same segment. A kernel density estimate is defined by

$$\tilde{f}(x) = \frac{1}{N} \sum_{i=1}^{N} K(x - x_i), \tag{13.13}$$

with

$$K(x) = |H|^{-0.5} K(H^{-0.5} x), \tag{13.14}$$

where N is the number of samples, and x_i stands for a sample from an unknown density function f. $K(\cdot)$ is the d-variate kernel function with compact support satisfying the regularity constraints, and H is a symmetric positive definite $d \times d$ bandwidth matrix. Usually, we have $K(x) = k_e(\phi)$, where $k_e(\phi)$ is a convex decreasing function, e.g. for a Gaussian kernel

$$k_e(\phi) = c_t e^{-\phi/2} \tag{13.15}$$

and for an Epanechnikov kernel,

$$k_e(\phi) = c_t \max(1 - \phi, 0) \tag{13.16}$$

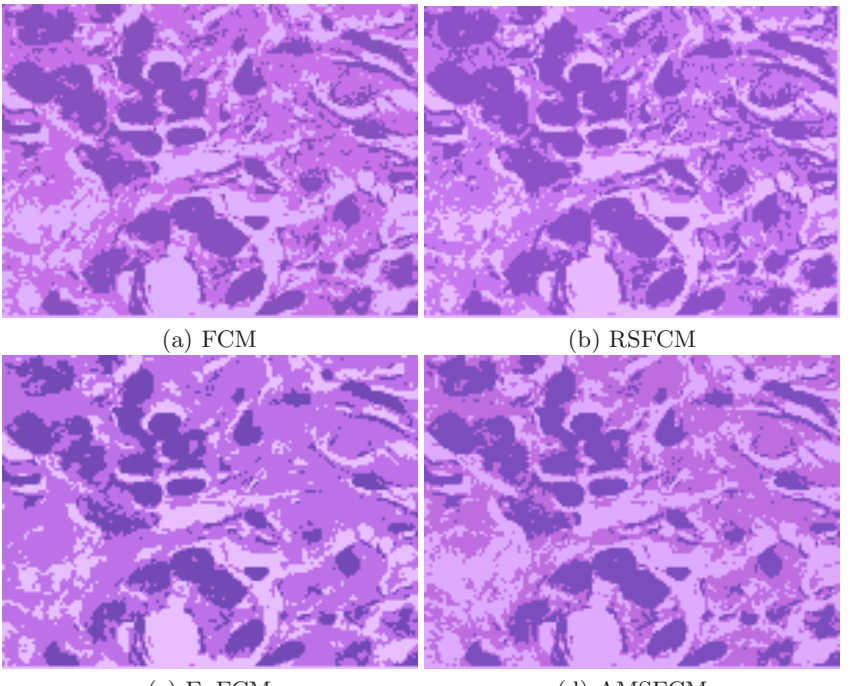
where c_t is a normalising constant.

If a single global spherical bandwidth is applied, $H = h^2\mathbf{I}$ ($\mathbf{I}$ is identity matrix), then we have

$$\tilde{f}(x) = \frac{1}{Nh^d} \sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right) \tag{13.17}$$

Since the kernel can be divided into two different radially symmetric kernels, we have the kernel density estimate as

$$\tilde{f}(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h^{\beta}(H_i^{\alpha})^q} k^{\alpha}(d(c_i^{\alpha}, x_i^{\alpha}, H_i^{\alpha})) k^{\beta}\left(||(c_i^{\beta} - x_i^{\beta})/(h^{\beta}(H_i^{\alpha}))||^2\right) \tag{13.18}$$



(a) FCM (b) RSFCM

(c) EnFCM (d) AMSFCM

Fig. 13.3. Segmentations of the image from Figure 13.2 using the four fuzzy c-means techniques with 3 clusters

where and α and β denote the spatial and temporal components respectively and $d(c_i^\alpha, x_i^\alpha, H_i^\alpha)$ is the Mahalanobis metric, i.e.

$$d(c_i^\alpha, x_i^\alpha, H_i^\alpha) = (x_i^\alpha - c_i^\alpha)^T H_i^{\alpha-1}(x_i^\alpha - c_i^\alpha). \tag{13.19}$$

Anisotropic mean shift is intended to modulate the kernels during the mean shift procedure. The objective is to keep reducing the Mahalanobis distance so as to group similar samples as much as possible. First, the anisotropic bandwidth matrix H_i^α is estimated with the following constraints:

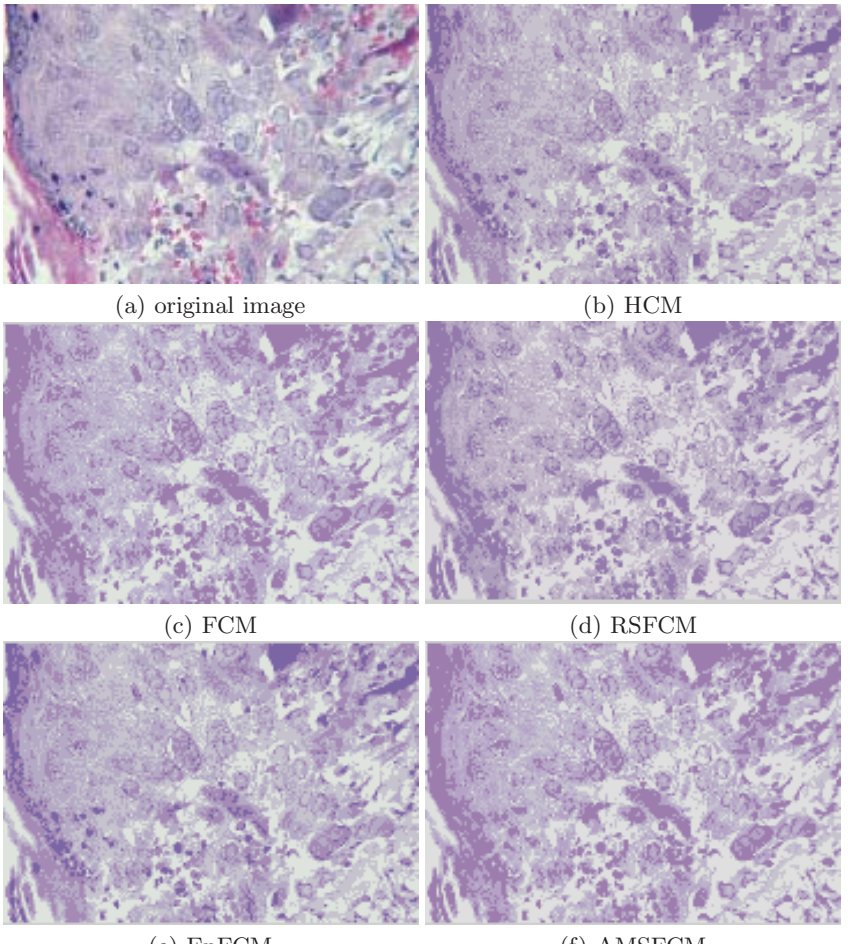
$$\begin{cases} k_e^\alpha(d(x, x_i, H_i^\alpha)) < 1 \\ k_e^\beta\left(||(x - x_i)/h^\beta(H_i^\alpha)||^2\right) < 1 \end{cases} \tag{13.20}$$

The bandwidth matrix can be decomposed to

$$H_i^\alpha = \lambda V A V^T \tag{13.21}$$

where λ is a scalar, V is a matrix of normalised eigenvectors, and A is a diagonal matrix of eigenvalues whose diagonal elements a_i satisfy

$$\prod_{i=1}^{p} a_i = 1 \tag{13.22}$$



(a) original image (b) HCM

(c) FCM (d) RSFCM

(e) EnFCM (f) AMSFCM

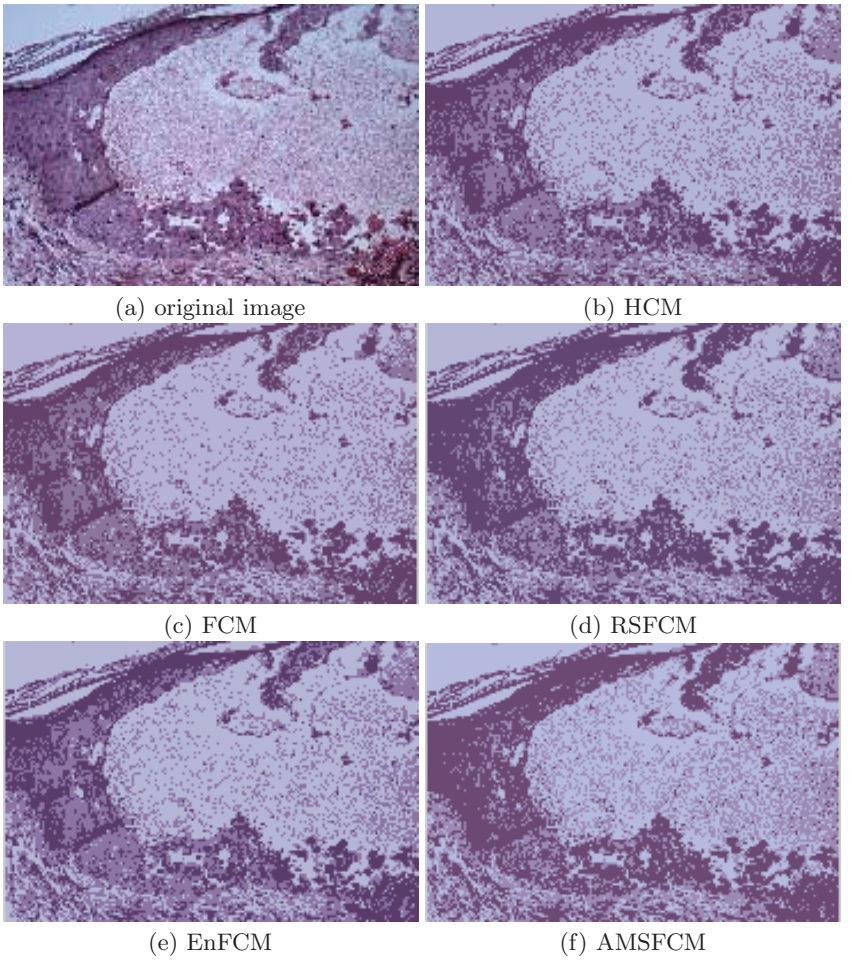
Fig. 13.4. Original *Herpes* image together with segmentations by all five c-means algorithms (based on 5 clusters)

The bandwidth matrix is updated by adding more and more points to the computational list: if these points are similar in intensity or colour, then the Mahalanobis distance will be consistently reduced. Otherwise, if the Mahalanobis distance is increased, these points will not be considered in the computation.

In our algorithm we combine fuzzy c-means and anisotropic mean shift segmentation. A significant difference between our approach and other similar methods is that our algorithm continuously inherits and updates the states, based on the mutual correction of FCM and mean shift.

Anisotropic mean shift based FCM (AMSFCM) proceeds in the following steps:

Step 1: Initialise the cluster centres c_i. Let $j = 0$.
Step 2: Initialise the fuzzy partitions μ_{ij} using Equation (13.3).



(a) original image (b) HCM

(c) FCM (d) RSFCM

(e) EnFCM (f) AMSFCM

Fig. 13.5. Original *Varicella* image together with segmentations by all five c-means algorithms (based on 3 clusters)

Step 3: Set $j = j + 1$ and compute c_i using Equation (13.4) for all clusters.
Step 4: Update μ_{ij} using Equation (13.3).
Step 5: For each pixel x_i determine anisotropic kernel and related colour radius using Equations (13.18) and (13.21). Note that mean shift is applied to the outcome image of FCM.
Step 6: Calculate the mean shift vector and then iterate until the mean shift, $M^+(x_i) - M^-(x_i)$, is less than a pixel considering the previous position and a normalised position change:

$$M^+(x_i) = \nu M^-(x_i) + (1 - \nu)\frac{\sum_{j=1}^{N}(x_j - M^-(x_i))||(M^-(x_i^\beta) - x_j^\beta)/(h^\beta H_j^\alpha)||^2}{\sum_{j=1}^{N}||(M^-(x_i^\beta) - x_j^\beta)/(h^\beta H_j^\alpha)||^2}$$

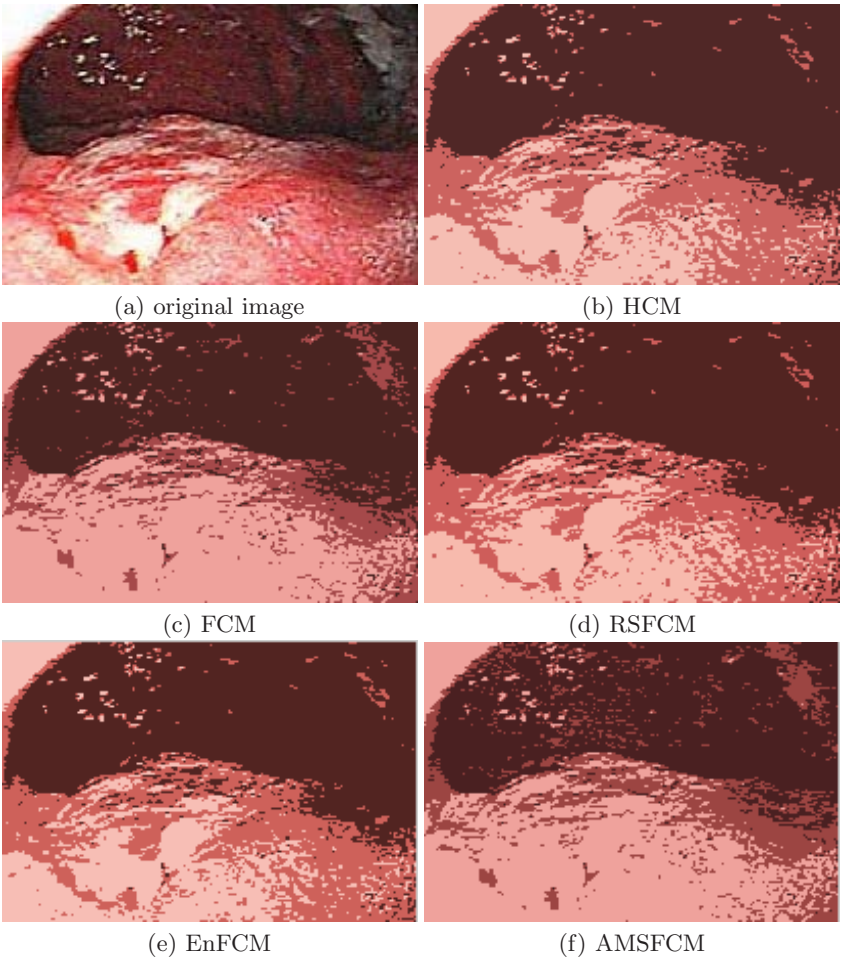
with $\nu = 0.5$.

Step 7: Merge pixels with similar colour.
Step 8: Repeat Steps 3 to 6 until convergence.

In Figure 13.3 we show the results of applying classical fuzzy c-means, the two improved algorithms (i.e., RSFCM and EnFCM), and our AMSFCM algorithm to the stained tissue image from Figure 13.2.

13.7 C-Means Based Segmentation of Medical Images

In this section we provide experimental results on series of medical images captured using different modalities. In particular we show segmentation results



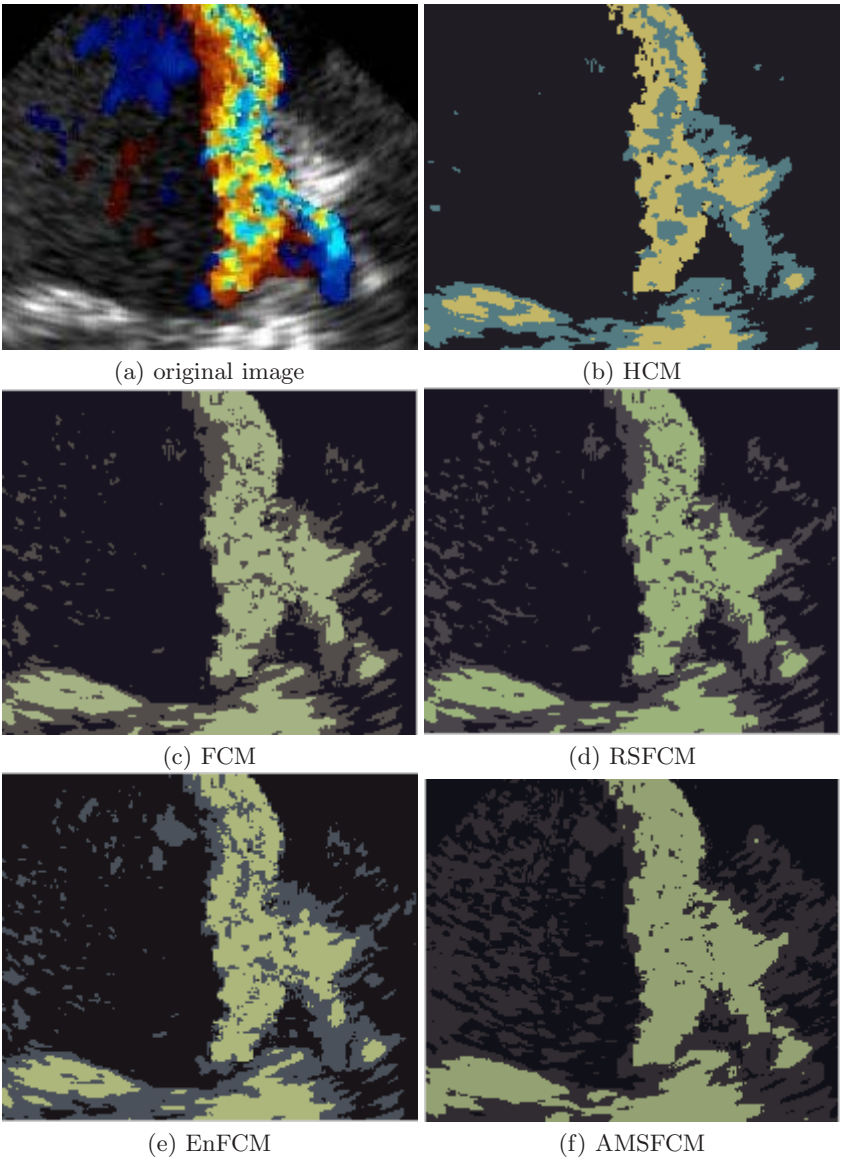
(a) original image (b) HCM

(c) FCM (d) RSFCM

(e) EnFCM (f) AMSFCM

Fig. 13.6. Original *Endoscopy* image together with segmentations by all five c-means algorithms (based on 3 clusters)

on four images: *Herpes* (Figure 13.4), *Varicella* (Figure 13.5), *Endoscopy* (Figure 13.6), and *Doppler* (Figure 13.7). For the *Herpes* image we used 5 clusters while for the other ones we used 3. For each image we show the original image together with the segmentations generated by all five algorithms.



(a) original image (b) HCM

(c) FCM (d) RSFCM

(e) EnFCM (f) AMSFCM

Fig. 13.7. Original *Doppler* image together with segmentations by all five c-means algorithms (based on 3 clusters)

As can be seen from the figures, in general the resulting segmentations share similarities. This is of course not surprising as they all share the common underlying concept of iteratively adjsting the cluster centres based on the generated partitions. Yet, it can also be observed that the fuzzy c-means methods typically provide improved results compared to the hard c-means images (in particular on the *Doppler* image). The three derivatives of fuzzy c-means provide similar performance but all three of them have a clear advantage over the conventional FCM algorithm in that they are by far more computationally efficient and run about 2-3 times faster than FCM [6]. As fast clustering is required in many real applications (e.g. health surveys and clinical diagnosis), the importance of this issue should be stressed.

13.8 Conclusions

In this chapter we have provided an overview of fuzzy c-means based image segmentation techniques and their application to medical imaging. Fuzzy c-means clustering provides a soft variant of the well known hard c-means algorithm. Various variants of this technique have been proposed in the literature and we have presented some of them including a novel technique based on a combination of fuzzy c-means with anisotropic mean shift segmentation. Fuzzy c-means segmentation is well established in medical imaging as has been demonstrated on a series of experiments. Nevertheless, it should be noted that apart from these techniques there is a wealth of other segmentation algorithms and that no single algorithm has been shown to work on every dataset. Rather, generic techniques as the ones covered here can provide the basis of more specialised approaches.

References

1. Ahmed, M., Yamany, S., Mohamed, N., Farag, A., Moriaty, T.: A modified fuzzy c-means algorithm for bias field estimation and segmentation of mri data. IEEE Trans. Medical Imaging 21, 193–199 (2002)
2. Bezdek, J.: A convergence theorem for the fuzzy isodata clustering algorithms. IEEE Trans. Pattern Analysis and Machine Intelligence 2, 1–8 (1980)
3. Bradley, P., Fayyad, U.: Refining initial points for k-means clustering. In: 15th Int. Conference on Machine Learning, pp. 91–99 (1998)
4. Cai, W., Chen, S., Zhang, D.: Fast and robust fuzzy c-means clustering algorithms incorporating local information for image segmentation. Pattern Recognition 40(3), 825–838 (2007)
5. Chen, S.C., Zhang, D.Q.: Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure. IEEE Trans. Systems, Man and Cybernetics - Part B: Cybernetics 34, 1907–1916 (2004)
6. Cheng, T., Goldgof, D., Hall, L.: Fast fuzzy clustering. Fuzzy Sets and Systems 93, 49–56 (1998)
7. Chuang, K., Tzeng, S., Chen, H., Wu, J., Chen, T.: Fuzzy c-means clustering with spatial information for image segmentation. Computerized Medical Imaging and Graphics 30, 9–15 (2006)

8. Comaniciu, D., Meer, P.: Mean shift analysis and applications. In: 7th Int. Conference on Computer Vision, pp. 1197–1203 (1999)
9. Comaniciu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. IEEE Trans. Pattern Analysis and Machine Intelligence 24, 603–619 (2002)
10. Dhillon, I., Guan, Y., Kogan, J.: Refining clusters in high dimensional text data. In: 2nd SIAM ICDM Workshop on clustering high dimensional data (2002)
11. Eschrich, S., Ke, J., Hall, L., Goldgof, D.: Fast accurate fuzzy clustering through data reduction. IEEE Trans. Fuzzy Systems 11, 262–270 (2003)
12. Haralick, R.M., Shapiro, L.G.: Image segmentation techniques. Computer Vision, Graphics, and Image Processing 29(1), 100–132 (1985)
13. Hartigan, J.: Clustering algorithms. John Wiley & Sons, New York (1975)
14. Hu, R., Hathaway, L.: On efficiency of optimization in fuzzy c-means. Neural, Parallel and Scientific Computation 10, 141–156 (2002)
15. Kass, M., Witkin, A.P., Terzopoulos, D.: Snakes: Active contour models. Int. Journal of Computer Vision 1(4), 321–331 (1988)
16. Kolen, J., Hutcheson, T.: Reducing the time complexity of the fuzzy c-means algorithm. IEEE Trans. Fuzzy Systems 10(2), 263–267 (2002)
17. Leski, J.: Toward a robust fuzzy clustering. Fuzzy Sets and Systems 137, 215–233 (2003)
18. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: 5th Berkeley Symposium on Mathematical statistics and probability, vol. 1, pp. 281–297 (1967)
19. Mao, J., Jain, A.K.: A self-organising network for hyperellipsoidal clustering (hec). IEEE Trans. Neural Networks 7(1), 16–29 (1996)
20. Szilagyi, L., Benyo, Z., Szilagyii, S.M., Adam, H.S.: MR brain image segmentation using an enhanced fuzzy c-means algorithm. In: 25th IEEE Int. Conference on Engineering in Medicine and Biology, vol. 1, pp. 724–726 (2003)
21. Wang, J., Thiesson, B., Xu, Y.-Q., Cohen, M.: Image and video segmentation by anisotropic kernel mean shift. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3022, pp. 238–249. Springer, Heidelberg (2004)
22. Zhang, B.: Generalized k-harmonic means dynamic weighting of data in unsupervised learning. In: 1st SIAM Int. Conference on Data Mining (2001)
23. Zhou, H., Schaefer, G., Shi, C.: A mean shift based fuzzy c-means algorithm for image segmentation. In: 30th IEEE Int. Conference Engineering in Medicine and Biology (2008)

14

Monitoring and Control of Anesthesia Using Multivariable Self-Organizing Fuzzy Logic Structure

J.S. Shieh[1], M.F. Abbod[2], C.Y. Hsu[3], S.J. Huang[4], Y.Y. Han[5], and S.Z. Fan[6]

[1] Department of Mechanical Engineering, Yuan Ze University, Taiwan
jsshieh@saturn.yzu.edu.tw
[2] School of Engineering and Design, Brunel University, West London, United Kingdom
Maysam.Abbod@brunel.ac.uk
[3] Department of Electrical Engineering, Yuan Ze University, Taiwan
[4] Department of Surgery, National Taiwan University Hospital, Taipei, Taiwan
[5] Department of Trauma, National Taiwan University Hospital, Taipei, Taiwan
[6] Department of Anesthesiology, College of Medicine, National Taiwan University, Taipei, Taiwan

Summary. In this chapter, the design and implementation of Self-Organizing Fuzzy Logic Controller (SOFLC) is explored with a particular application to control a multivariable model of anesthesia. A concept called decomposition of multivariable self-organizing fuzzy logic structure is proposed in this chapter. Hence, the basic forms of a simple 2 terms SOFLC to a multi-term complex multi-input/multi-output (MIMO) controller will be presented. Different design strategies of MIMO will be outlined and the application of SOFLC systems to muscle relaxation and depth of anesthesia control will be explored in the simulations. After comparison with four different MIMO controllers, the successful simulation results have given confidence to perform on-line clinical trials at the operating theatre in the near future.

14.1 Introduction

Control of non-linear systems has grown rapidly due to the fact that most systems are inherently non-linear, moreover, linear control systems can only perform well on a linearized model of the process around the operating points. Most controllers such as a PID three term controllers, Model-Based Predictive Control (MBPC) and robust control (H_α) can do very well on a fixed set point. However, in recent years, there has been a move towards intelligent control with a qualitative dimension due to the widespread dissatisfaction with quantitative systems engineering. One of the main attractions of intelligent system design is the possibility of multivariable control system without the need for extensive dynamic models of the process [11, 25]. The main difficulty in the multivariable case is the interaction between variables together with sensitivity to faults in various channels. Intelligent systems, such as Neural Networks (NN), Fuzzy Logic Control (FLC), and Genetic Algorithms (GA), have been at the forefront

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 273–295.
springerlink.com

of such methodologies and have proved to be strong contenders for other forms of control [8].

The application of intelligent control to medical systems has been around for many years [5, 14, 19], but due to the nature of the humans, differences from one person to another, the dynamic changes in the human response to external stimuli's and the effect of the different drugs on patients, a form of an adaptable intelligent controller can fix the description very well. An attractive approach to solving these problems is provided by the self-organizing fuzzy logic controller (SOFLC), which was first proposed by Procyk and Mamdani [26]. By mimicking the human learning process, the SOFLC has a learning algorithm and is capable of generating and modifying control rules according to an evaluation of the system's performance. There have been many studies and applications of SOFLC in recent years, but only a few in biomedical systems. Linkens and Hasnain [15] published an early study on SOFLC of muscle relaxation, but only in computer simulations. Recently, this has been implemented in clinical trials in muscle relaxation [22, 27], depth of anesthesia [34, 35], and pain control in patient controlled analgesia [30]. However, most of these applications are dealing with two inputs and one output. When we meet the multivariable self-organizing fuzzy logic structure, it was found that there are still some problems with the SOFLC algorithm after many applications in multivariable structure, mostly in its difficulty to handle the performance index and rule-base in multidimensional space. An idea stimulated by the decomposition of multivariable control rules of fuzzy system into a set of one-dimensional systems led Gupta et al. [9] to a solution of multivariable fuzzy systems. A concept called the decomposition of multivariable self-organizing fuzzy logic structure is presented in this chapter. In Section 14.2 the generic multivariable self-organizing fuzzy logic structures are presented and some formal properties of the structures are discussed. Simulation of anesthesia system either in two-input / two-output or four-input / two-output for SOFLC structures are demonstrated in Section 14.3. Finally, the concluding remarks are given in Section 14.5.

14.2 Multivariable Self-Organizing Fuzzy Logic Structure

Recently research on the application of fuzzy set theory to the design of biomedical control systems has led to interest in the theory and description of the multivariable structure of these systems due to two vital factors. One is real biomedical control systems are multidimensional, and another is the computer implementation of these biomedical systems requires the processing of a huge data base due to the complexity of human being. Therefore, the analysis and design procedures for such systems are consequently very difficult. In search of previous study of SOFLC structure, most of these applications are dealing with two inputs and one output as shown in Figure 14.1.

As an explanatory example, take as the starting point a simple two-input / one-output self-organizing fuzzy logic structure illustrated in Figure 14.2. SOFLC is an extension of a simple fuzzy logic controller with the self-organizing

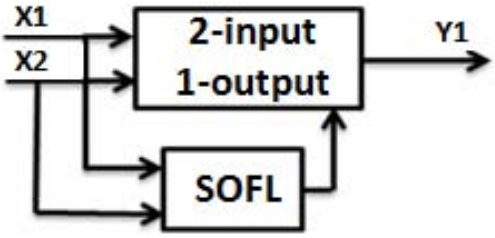


Fig. 14.1. Self-organizing fuzzy logic structure for 2-input / 1-output

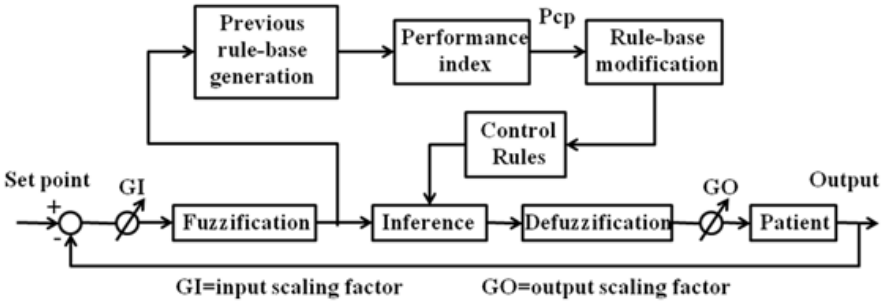


Fig. 14.2. Design of drug controller using a SOFLC algorithm

level that incorporates four new functional blocks: (1) the previous rule-base generation, (2) the performance index, (3) the rule-base modification algorithm, and (4) the control rule-base performance measure.

14.2.1 The Previous Rule-Base Generation

This rule-base can be generated either from expert experience (i.e., medical doctors) or from learning input and output data. Hence, the previous rule-base may have some rules to start with if it begins from expert experience, or may have no rules initially if it starts from zero knowledge. However, after introducing several data into the process, the previous rule-base will be modified by current input and output data. In this chapter, the initial rule-base is generated from simple FLC based on the try-and-error method from our researcher which was expert in fuzzy logic control but only had a little knowledge of anesthesia system.

14.2.2 The Performance Index

The performance index measures the deviation from the desired response and calculates the appropriate changes that are required in the output of the controller. The generation and modification of the control rules is achieved by assigning a credit or reward value to the individual control actions that make a major contribution to the present performance. The credit value is obtained from a fuzzy algorithm which defines the desired performance linguistically and has the

Error	Change in error						
	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NM	NM	NS	ZE
NM	NB	NB	NM	NM	NS	ZE	NS
NS	NB	NB	NS	NS	ZE	PS	PM
ZE	NB	NM	ZE	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PS	PB	PB
PM	NS	ZE	PS	PM	PM	PB	PB
PB	ZE	PS	PM	PM	PB	PB	PB

PB: positive big
PM: positive medium
PS: positive small
ZE: zero
NS: negative small
NM: negative medium
NB: negative big

Fig. 14.3. SOFLC performance index matrix [15]

same form as the control algorithm of the generic fuzzy logic controller. Hence, these linguistic performance rules are derived from a qualitative "feel" for the process and are intended to provide fast convergence around the equilibrium state to achieve high accuracy. For this reason, it is not specific to the type of process being controlled. In other words, this performance index may be very similar for different processes. In this work the performance index was derived from previous research work [15] as shown in Figure 14.3.

14.2.3 The Rule-Base Modification Algorithm

The rule modification procedure can be explained assuming that a process has a time-lag of m samples. If the present instant is nT, this means that the control action at sample (nT-mT) has contributed most to the process performance at the sampling instance nT. Thus, the original implication:

$$E(nT - mT) \rightarrow CE(nT - mT) \rightarrow U(nT - mT)$$

should be changed to:

$$E(nT - mT) \rightarrow CE(nT - mT) \rightarrow U(nT - mT) + P_o(nT)$$

where E and CE are error and change-in-error from the set-point respectively; U is the controller output; P_{cp} is the correction issued by the performance index.

After the rule modification procedure has taken place, a new rule is generated from the input and output data of the controller at each sampling step. The method of logic examination [38] can be employed to obtain the new rules. If the new generated rule has no match in the rule-base, it will be added to rule-base. However, if it already exists in the rule-base, it will be replaced.

14.2.4 The Control Rule-Base Performance Measure

After modifying the three functional blocks, the control rule-base becomes accurate (i.e. no noise contamination and conflicting rules). If the performance of the controller is satisfied by the necessary criteria which are strongly dependent on individual system requirement, the rule-base of the controller will stop modification and the rule-base will converge to a constant rule-base.

However, the multivariable self-organizing fuzzy logic structure as shown in Figure 14.4, still have some problems with the structure when applied to multivariable systems, mostly in its difficulty to handle the performance index and rule-base in multidimensional space. An idea stimulated by the decomposition of multivariable control rules of fuzzy system into a set of one-dimensional systems led Gupta et al. [9] to a solution of multivariable fuzzy systems. A concept called the decomposition of multivariable self-organizing fuzzy logic structure is shown in Figure 14.5 as a simple example for this 3-input and 1-output SOFLC. Furthermore, it is easy to extend this concept to decompose 2-input / 2-output and 3-input / 2-output SOFLC structure in Figure 14.6.

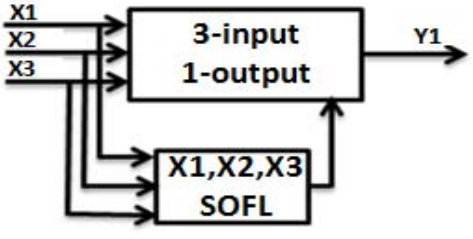


Fig. 14.4. Conventional self-organizing fuzzy logic structure for 3-input / 1-output

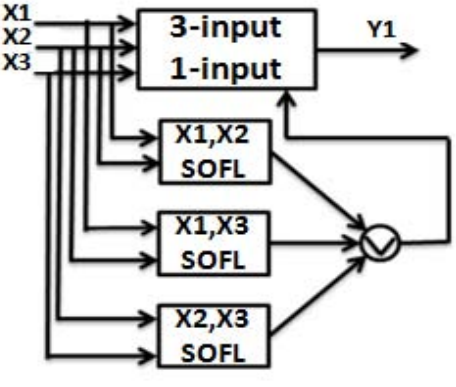


Fig. 14.5. Decomposition of self-organizing fuzzy logic structure for 3-input / 1-output

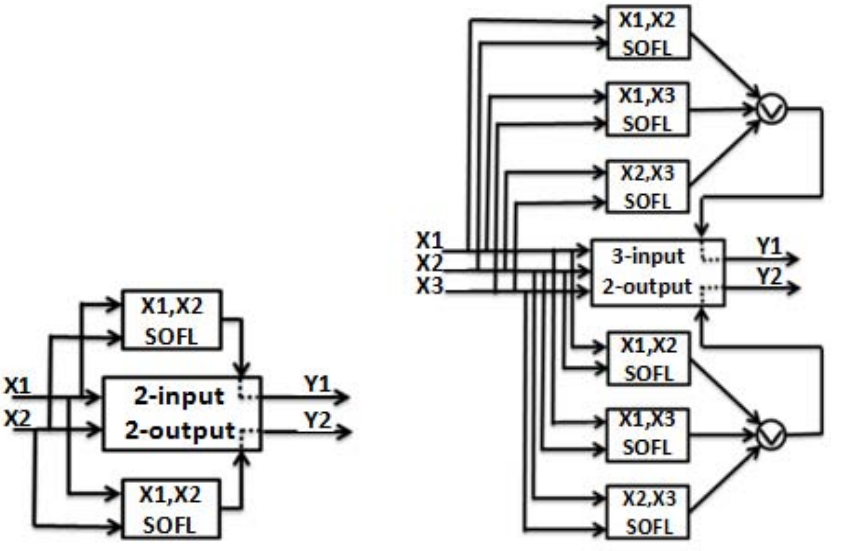


Fig. 14.6. Decomposition of self-organizing fuzzy logic structure for 2-input / 2-output and 3-input / 2-output

14.3 Simulation of Anesthesia for Multivariable SOFLC Structures

14.3.1 Simulation Methods

Anesthesia is the art or science of removing sensation of, and reaction to, a surgical procedure. Anesthesia means loss of all sensation whether it is a sense of pain, touch, temperature or position [1]. Modern general anesthesia comprises the triad of muscle relaxation, unconsciousness, and analgesia (i.e. pain relief). Each of these conditions has been considered in recent years as possible scenarios for automated drug infusion via feedback strategies. The major roles performed by a clinical anesthetist are the maintenance of drug-induced muscle relaxation, unconsciousness and analgesia. During the last two decade, the application of simple control (i.e., PID) and advanced control (e.g., adaptive & intelligent) techniques to drug-induced muscle relaxation and unconsciousness in operating theatre has been investigated [3, 13, 16, 20, 21, 31, 32, 34, 35, 36, 39]. The main problem in drug-induced unconsciousness is to measure clinical signs which can be used on-line to the system. The measurement of muscle relaxation is considerably easier via evoked electromyogram (EMG) responses using commercial instruments such as a Datex Relaxograph. Hence, these EMG responses are still as the most general reliable guide for administering intravenous (e.g., atracurium, cisatracurium, or rocurocium) of muscle relaxation control.

However, depth of anesthesia (i.e. unconsciousness) is much harder to define and not readily measurable. In practice, anesthetists have a number of clinical signs and on-line measurements which can be used selectively for the determination of the patient's state. Therefore, many methods have been used for feedback control of anesthetic depth based on different clinical measurements, such as blood pressure [23, 29, 40], electroencephalograph (EEG) signals [33], minimum alveolar concentration (MAC) values [37], plasma concentration of propofol [28] and auditory evoked response (AER) [6, 7]. However, anesthetists still use blood pressure as the most general reliable guide for administering intravenous (e.g., propofol) or inhalational anesthetics (e.g., isoflurane, desflurane, or sevoflurane). The measurement of pain is the hardest of all, since it is heavily subjective, and liable to many levels of personal interpretation. Generally speaking, analgesia is mainly concerned with postoperative conditions. It will not be considered in the simulation of this chapter.

It is considered that the major roles performed by a clinical anesthetist are the maintenance of drug-induced muscle relaxation, unconsciousness, and analgesia. Anesthetic drugs with a rapid onset and short duration of action are highly desirable. The more anesthetists understand the drug's features accurately, the more patient's safety was protected. Pharmacology, the basic for using closed-loop control, consists of two main categories known as pharmacokinetics (PK) and pharmacodynamics (PD). Pharmacokinetics is the study of the concentration of drugs in tissue as a function of time and dose schedule, where as pharmacodynamics is the study of the relationship between drug concentration and effect. Therefore, not only can the anesthetist improve their anesthetic skills through many clinical trials but also by means of pharmacology. In this section of simulation, we use the most common drugs in modern surgery of atracurium for controlling muscle relaxation and isoflurane for controlling blood pressure via their PK-PD compartment models as shown in Figure 14.7 [4].

14.3.2 The Atracurium Mathematical Model

Pharmacokinetics

According to previous studies [17, 18], the drug pharmacokinetics can be expressed by the following equation:

$$G_1(s) = \frac{9.94(1 + 10.64s)}{(1 + 3.08s)(1 + 34.42s)} \tag{14.1}$$

Equation (14.1) describes the pharmacokinetics of the muscle relaxation system relating to the drug atracurium.

Pharmacodynamics

Similarly, to characterize different aspects of drug effect a hypothetical effect compartment is introduced in the above structure (Figure 14.7) leading to the following transfer function:

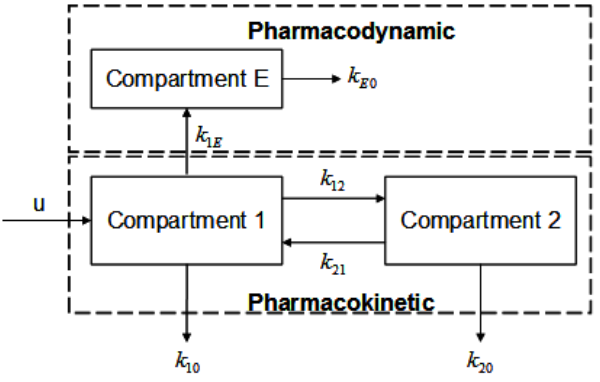


Fig. 14.7. Traditional pharmacological patient model, 2-compartment for pharmacokinetic model and one compartment model for pharmacodynamics, u is system input. The parameters of k_{10} and k_{20} are elimination paths based on Hofmann elimination criteria, k_{12} and k_{21} are first-order rate constants associated with the movement of drug from compartment 1 to compartment 2 and compartment 2 to compartment 1, respectively, and k_{E0} is the rate constant for elimination from the effect compartment.

$$G_{11}(s) = \frac{K_1(1+T_4 s)e^{-\tau_1 s}}{(1+T_1 s)(1+T_2 s)(1+T_3 s)} \tag{14.2}$$

where τ_1= 1 min, $K_1 = 1$, T_1= 4.81 min, T_2= 34.42 min, T_3= 3.08 min, T_4= 10.64 min. Moreover, the following non-linearity represented by a Hill equation is used to relate the effect to a specific drug concentration:

$$E_{eff} = E_{\max}\frac{X_E^{\alpha}}{X_E^{\alpha} + (X_E(50))^{\alpha}} \tag{14.3}$$

where X_E is the drug concentration, α the power and $X_E(50)$ the drug concentration at 50% effect with the following values**:** $E_{max} = 100\%$, $X_E(50) = 0.404$ $\mu g\ ml^{-1}$, $\alpha = 2.98$.

14.3.3 The Isoflurane Unconsciousness Model

There is no doubt that anything that is related to the human brain represents a very complex entity, and anesthesia or unconsciousness which affects the brain has indeed been the subject of many conflicting views. Hence, depth of anesthesia (i.e. unconsciousness) is much harder to define and not readily measurable. In practice, anesthetists have a number of clinical signs and on-line measurements which can be used selectively for the determination of the patient's state. Routinely, anesthetists still use blood pressure as the most general reliable guide for inhalational anesthetics (e.g., isoflurane, desflurane, or sevoflurane) or administering intravenous (e.g., propofol). Hence, in studies conducted by previous groups [24], step responses to changes in inspired concentration of isoflurane

from a vaporizer were performed. If the changes in inhaled isoflurane concentration are small (i.e., less than 5 %), the responses could be approximated by linear characteristics. However, if the changes do not fall within this range, the responses are in general non-linear and time-varying. Thus, a first-order linear model with dead-time has been adopted, having a time-constant of 1-2 minutes. The magnitude of the time-constant is long enough to absorb some inaccuracy of dead-time estimate due to breathing variation. On the other hand, in order to estimate the steady-state gain, it is assumed that a relatively sensitive patient needs 2 % isoflurane for a 30 mmHg reduction in MAP. Therefore, the model describing variations of blood pressure to small changes in inhaled isoflurane concentration can be written as:

$$G_{22}(s) = \frac{\Delta MAP(s)}{U_2(s)} = \frac{K_2 e^{-\tau_2 s}}{(1 + T_5 s)} \tag{14.4}$$

where τ_2= 0.42 min, T_5= 2 min, K_2= -15 mmHg/percent.

14.3.4 Interactive Component Model

Regarding atracurium to blood pressure interaction [18], this has been investigated in human beings and there seems to be a small increase in heart rate when atracurium is administered. As an initial approximation, therefore, this pathway has been ignored in the dynamic model. However, it should be noted that this may not be approximate for other drugs for unconsciousness, such as other inhalational drugs, such as desflurane or sevoflurane. On the other side, the interaction of isoflurane to muscle relaxation is small but significant. An experiment was performed by Dr Asbury in 1990, in which a patient of 47 without a kidney but having a renal transplant was anaesthetized. Step changes of $0 \sim 1$ % isoflurane infusions were superimposed on steady relaxation levels achieved 50 minutes into the operation via atracurium infusion. Transient responses for both on and off conditions were obtained, and dynamics estimated for each case. Because there was not a large difference between the phases, an averaged transfer function was obtained as follows:

$$G_{12}(s) = \frac{K_4 e^{-\tau_4 s}}{(1 + T_6 s)(1 + T_7 s)} \tag{14.5}$$

where τ_4= 1 min, T_6= 2.83 min, T_7= 1.25 min, K_4= 0.27 .

14.3.5 The Overall Multivariable Anesthetic Model

From the previous sections description, the overall linear multivariable system combining muscle relaxation (i.e., paralysis) together with unconsciousness (in terms of blood pressure measurements) can be summarized by the following system:

$$\begin{bmatrix} Paralysis \\ \Delta MAP \end{bmatrix} = \begin{bmatrix} G_{11}(s) & G_{12}(s) \\ 0 & G_{22}(s) \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} \tag{14.6}$$

where

$$G_{11}(s) = \frac{1.0e^{-s}(1+10.64s)}{(1+3.08s)(1+4.81s)(1+34.42s)}$$

$$G_{12}(s) = \frac{0.27e^{-s}}{(1+2.83s)(1+1.25s)}$$

$$G_{22}(s) = \frac{-15.0e^{-0.42s}}{(1+2s)}$$

Finally, the overall non-linear multivariable system combining all the effects is obtained by including the non-linearity of pharmacodynamics of atracurium drug only since isoflurane drug-effect is considered to reflect linear characteristics within a range already specified in the preceding sections.

14.4 Simulation Results

In order to demonstrate the performance of the proposed multivariable SOFLC structure, the model has been taken the equation (14.6) for simulation this two-input (i.e., muscle relaxation error and blood pressure error), and two-output (i.e., atracurium and isoflurane), anesthesia control system using fuzzy logic and self-organizing fuzzy logic structures. Moreover, in order to reduce the steady-state errors, the error integrations of muscle relaxation and blood pressure have been considered for simulation this four-input (i.e., muscle relaxation error, muscle relaxation integration error, blood pressure error, and blood pressure integration error), and two-output (i.e., atracurium and isoflurane), anesthesia control system using fuzzy logic and self-organizing fuzzy logic structures as well. Hence, four kinds of fuzzy logic structures have been considered in this work, as described in the following.

14.4.1 Fuzzy Logic Control of Two-Input and Two-Output Anesthesia System

Figure 14.8 shows the FLC closed-loop control structure of the two-input and two-output anesthesia system. Control rules, membership functions, fuzzy inference engine and defuzzification are the essential elements in the fuzzy logic control. To perform fuzzy inference and describe this FLC control system, we chose two inputs which were the error of muscle relaxation (i.e., M_e) and the error of blood pressure (i.e., B_e) and two outputs which were the atracurium infusion rate (i.e., Atra_Inf) and isoflurane concentration (i.e., Iso_Conc). So, the fuzzy logic structure for this two-input and two-output is shown in Figure 14.9.

In order to fuzzify the inputs and output, the error of muscle relaxation (M_e) and the error of blood pressure (B_e) were divided into seven levels, namely negative big (NB), negative medium (NM), negative small (NS), zero (ZE), positive small (PS), positive medium (PM), and positive big (PB). The change of atracuriun infusion (Atra_Inf) and isoflurane concentration (Iso_Conc) were divided into four levels, namely zero (ZE), positive small (PS), positive

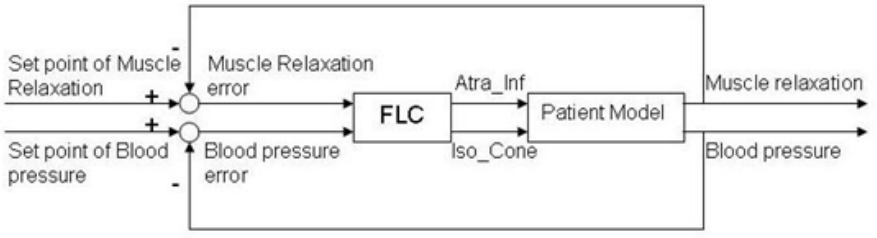


Fig. 14.8. Closed loop FLC system

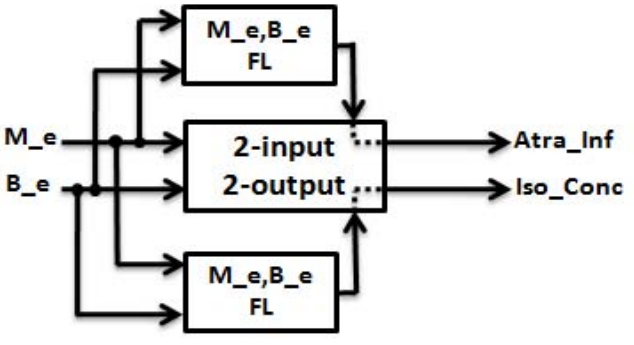


Fig. 14.9. The fuzzy logic structure for this two-input / two-output

Atra_Inf	NB	NM	NS	ZE	PS	PM	PB
NB	PB		PS		ZE		ZE
NM		PM		ZE		ZE	
NS	PB		PS		ZE		ZE
ZE		PM		ZE		ZE	
PS	PM		ZE		ZE		ZE
PM		PM		ZE		ZE	
PB	PM		ZE		ZE		ZE

(a)

Iso_Conc	NB	NM	NS	ZE	PS	PM	PB
NB	ZE		ZE		ZE		ZE
NM		ZE		ZE		ZE	
NS	ZE		ZE		ZE		ZE
ZE		ZE		ZE		ZE	
PS	PS		PS		PS		ZE
PM		PM		PS		ZE	
PB	PB		PM		PS		PS

(b)

Fig. 14.10. The rule-bases of two-input and two-output for fuzzy logic (a) Atracurium rule-base (b) Isoflurane rule-base

medium (PM), and positive big (PB). There is no negative fuzzy set because the absolute output values of atracurium infusion and isoflurane concentration were used in this simulation so there are no negative values of these infusion rate and concentration. There are many shapes of possible membership functions, such as triangle and trapezoid, which can be used in the fuzzy logic

controller. In this study, a triangular shape is used and a 25% overlap for contiguous fuzzy sets is reckoned [12] for two inputs (M_e and B_e), and two outputs (Atra_Inf and Iso_Conc). A try-and-error method was adopted to generate the initial rule-base; this method is based on good knowledge of fuzzy logic but less in anesthesia. Twenty five rules were developed to control the system as shown in Figure 14.10. In this simulation, the set points of muscle relaxation and blood pressure were set to 80% paralysis and 110 mmHg respectively. Each simulation was performed for 150 min surgical operation. The simulation results are shown in Figure 14.11. Unfortunately, these initial rules gave poor control of the muscle relaxation and blood pressure where some steady state errors occurred. Therefore the rule-bases need to be modified due to the poor designed rule which were generated by a non expert in anesthesia. Hence, this could be done using a SOFLC algorithm to further fine-tune the rule-bases.

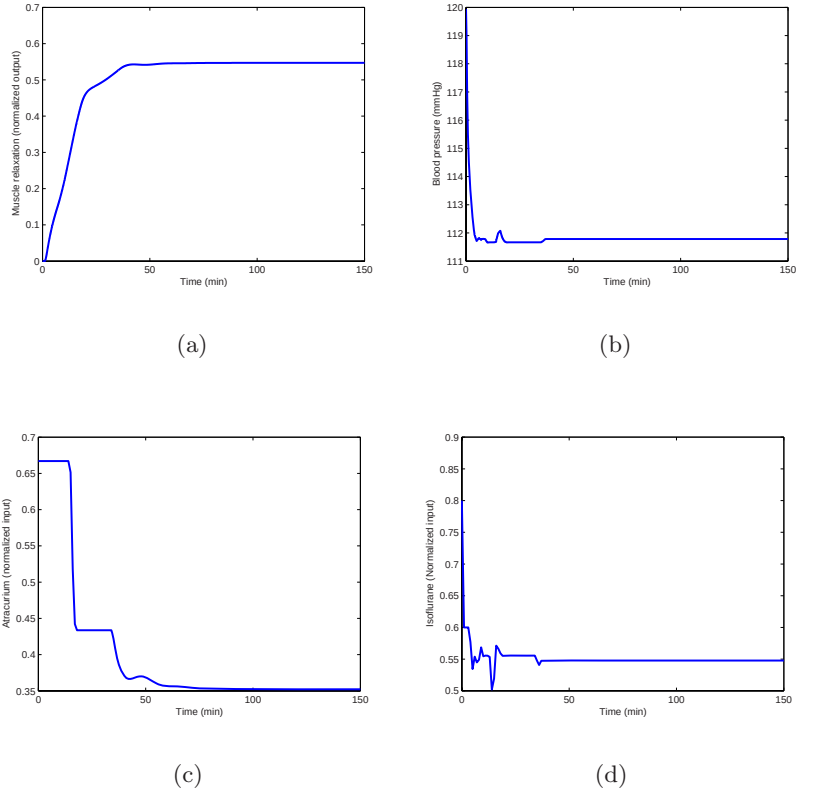


Fig. 14.11. The simulation of two-input and two-output anesthesia system using fuzzy logic (a) Muscle relaxation output (b) Blood pressure output (c) Atracurium input (d) Isoflurane input

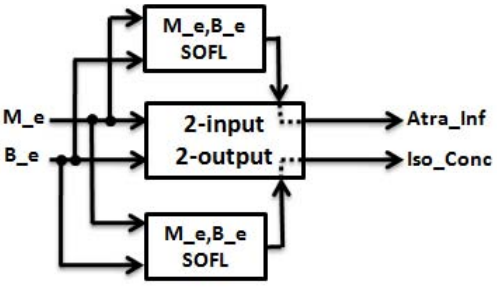


Fig. 14.12. The two-input and two-output anesthesia system using SOFLC structure

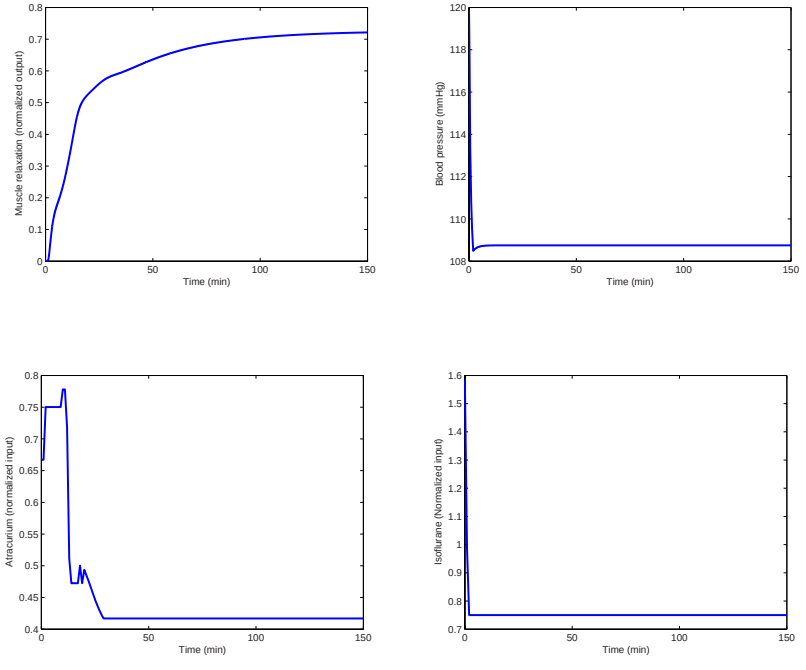


Fig. 14.13. The simulation of two-input and two-output anesthesia system using SOFLC (a) Muscle relaxation output (b) Blood pressure output (c) Atracurium input (d) Isoflurane input

14.4.2 Self-Organizing Fuzzy Logic Control of Two-Input and Two-Output Anesthesia System

SOFLC is a two-level hierarchical controller. The basic level is a simple fuzzy logic controller, while the second level is a self-organizing level that supervises the basic level by monitoring its performance, subsequently generating and modifying the control rules. Hence, we applied this SOFLC in multivariable structure of two-input and two-output anesthesia simulation system. The SOFLC

Atra_Inf	NB	NM	NS	ZE	PS	PM	PB
NB	PB		PS		ZE		ZE
NM		PM		ZE		ZE	
NS	PB		PS		ZE		ZE
ZE	*PB*	*PB*	*PM*	*PM*		ZE	
PS	PM		ZE		ZE		ZE
PM	*PB*	PM		ZE		ZE	
PB	PM		ZE		ZE		ZE

(a)

Iso_Conc	NB	NM	NS	ZE	PS	PM	PB
NB	ZE		ZE		ZE		ZE
NM		ZE		ZE		ZE	
NS	ZE		ZE		ZE		ZE
ZE	*PS*	*PS*	*PS*	*PS*		ZE	
PS	*PM*		PS		PS		ZE
PM	*PM*	PM		PS		ZE	
PB	PB		PM		PS		PS

(b)

Fig. 14.14. The rule-bases of two-input / two-output for SOFLC (a) Atracurium rule-base (b) Isoflurane rule-base (Notation: The italic and underline rules in the atracurium and isoflurane rule-bases represented generated from self-organized fuzzy logic algorithm)

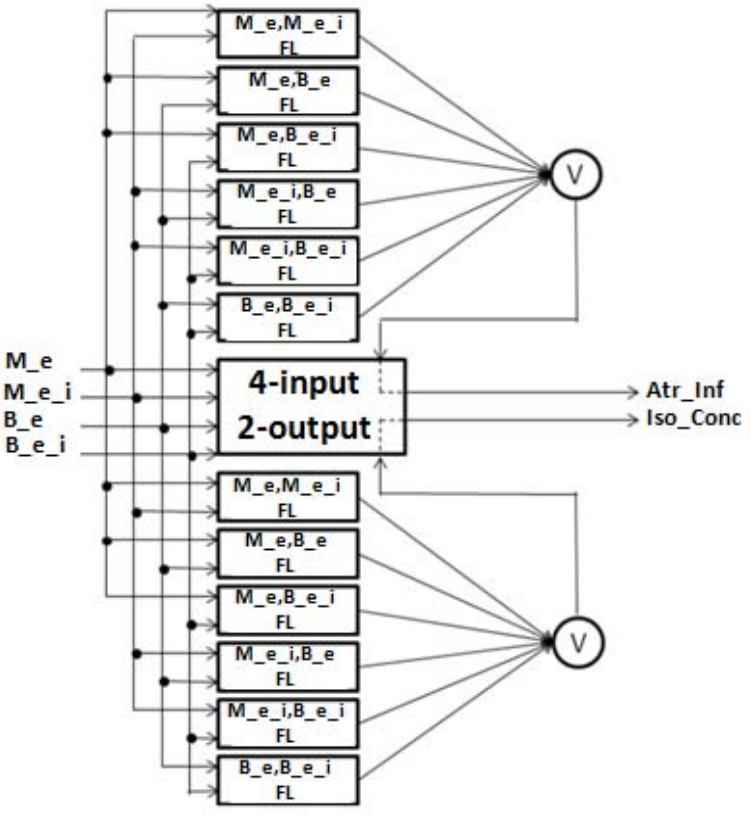


Fig. 14.15. The fuzzy logic structure for this four-input / two-output

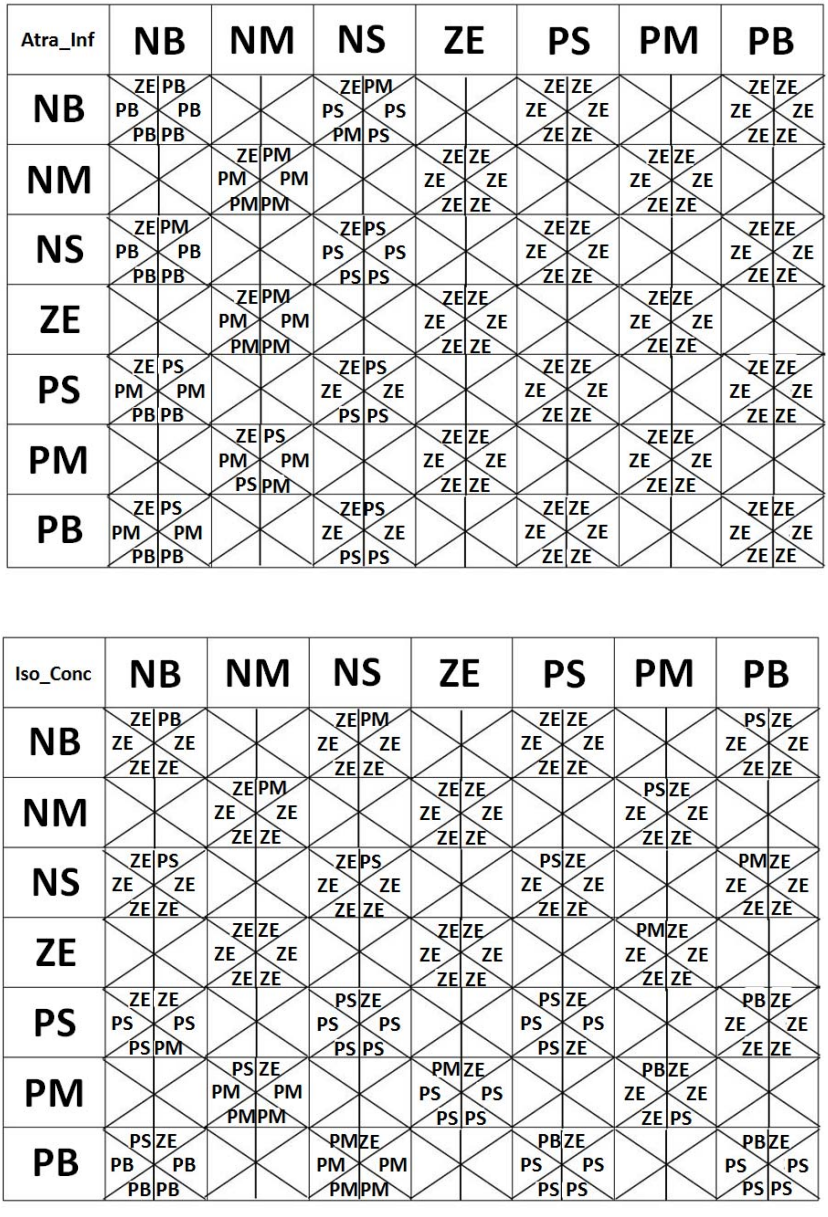


Fig. 14.16. The rule-bases of four-input / two-output for fuzzy logic (a) Atracurium rule-base (b) Isoflurane rule-base

structure was shown in Figure 14.12. In order to compare it to a basic fuzzy logic controller, the set points and partition of fuzzy sets for the inputs and outputs were the same used in the previous method. Also, in this simulation,

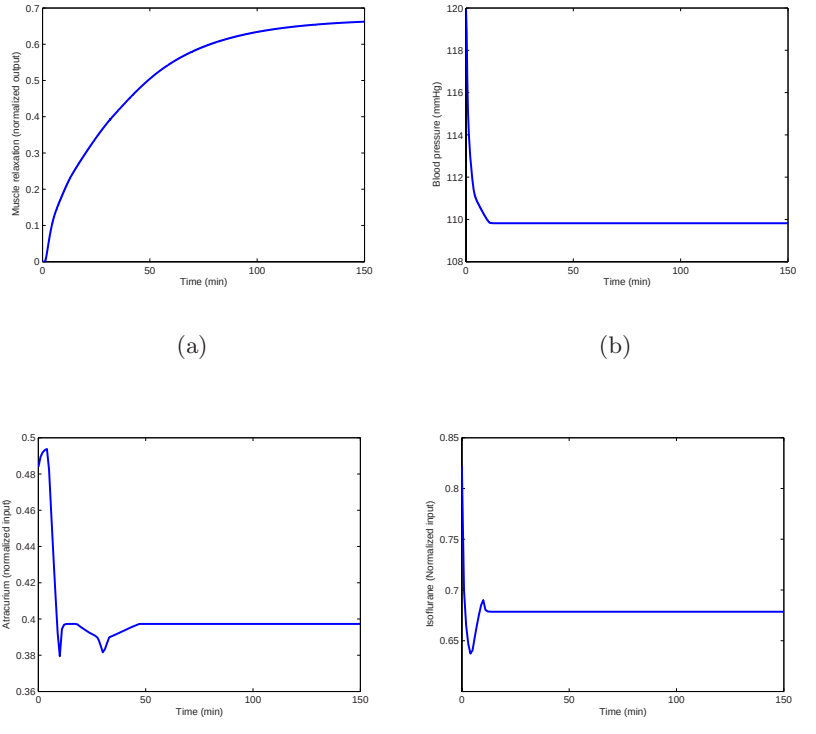


Fig. 14.17. The simulation of two-input and two-output anesthesia system using fuzzy logic (a) Muscle relaxation output (b) Blood pressure output (c) Atracurium input (d) Isoflurane input

the initial rule-base (i.e., 25 rules) was taken from the previous basic FLC. The simulation results are shown in the Figure 14.13. The controller performance is better than the previous method but the steady state error still dominant in the outputs although some rules were generated by self-organizing fuzzy logic structure as shown in Figure 14.14. Therefore, the control structure needs to be modified in order to overcome the steady state error problems.

14.4.3 Fuzzy Logic Control of Four-Input/Two-Output Anesthesia System

In order to reduce the steady state error, an integration of the error was considered as an input to the system. Hence, four inputs were defined as the error of muscle relaxation (M_e), the integration error of muscle relaxation (M_e_i), the error of blood pressure (B_e), and the integration error of blood pressure (B_e_i). Whereas the two outputs were the same as in the previous method, namely atracurium

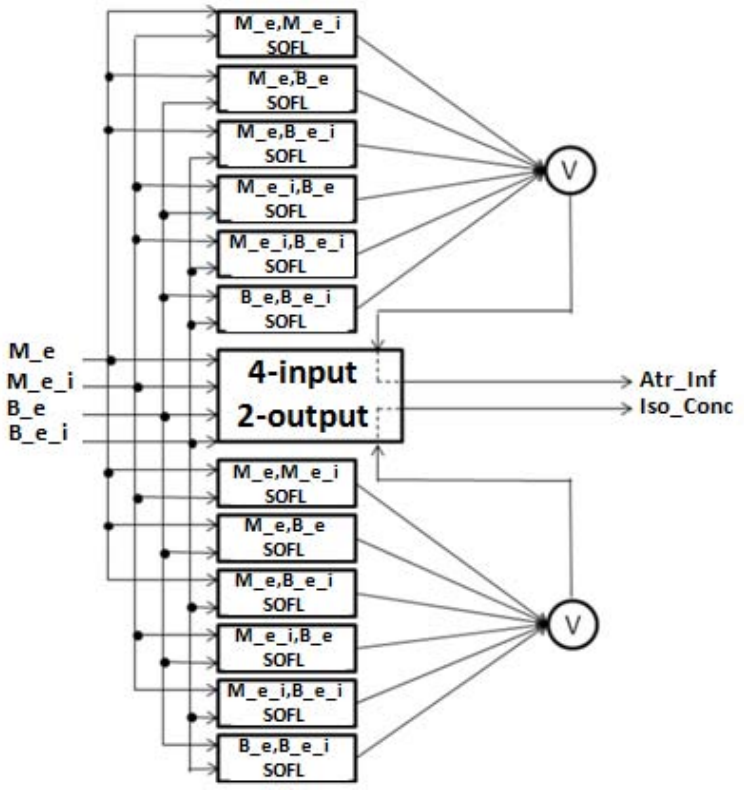


Fig. 14.18. The SOFLC structure for four-input / two-output structure

Table 14.1. The steady state errors of muscle relaxation and blood pressure of these four methods

	FLC (2-input / 2-output)	SOFLC (2-input / 2-output)	FLC (4-input / 2-output)	SOFLC (4-input / 2-output)
Muscle Relaxation Steady-state error	-0.25293	-0.078626	-0.13776	-0.006190
Blood Pressure Steady-state error	1.7836	-1.25	-0.17889	0.065204

infusion rate (Atra_Inf) and isoflurane concentration (Iso_Conc). The controller structure (four-input / two-output) is shown in Figure 14.15. According to try-and-error method, the designer (expert in fuzzy logic control but only had a little

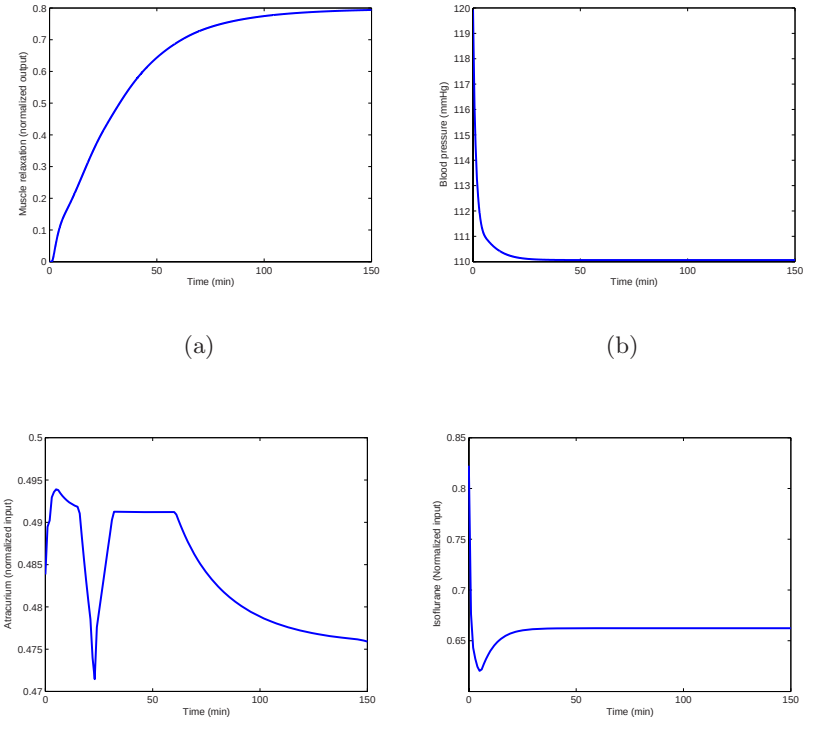
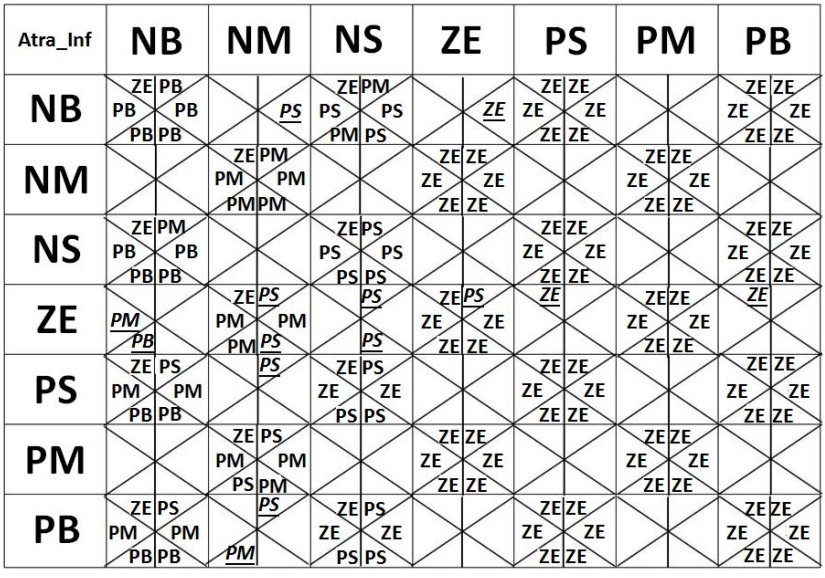


Fig. 14.19. The simulation of four-input / two-output anesthesia system using SOFLC (a) Muscle relaxation output (b) Blood pressure output (c) Atracurium input (d) Isoflurane input

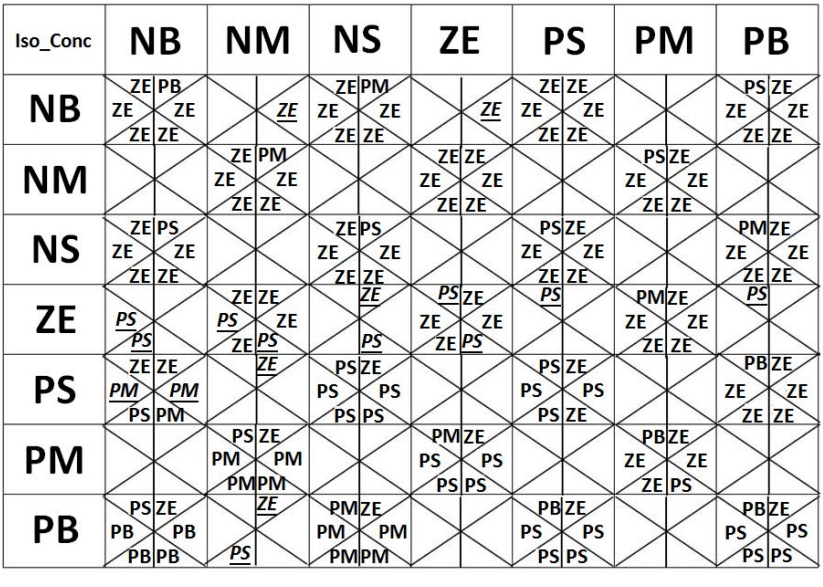
knowledge of anesthesia system) has generated both rule-bases for controlling the atracurium infusion and isoflurane concentration. Six fuzzy rule-bases were developed; each has twenty five rules as shown in Figure 14.16. In order to compare with the two-input / two-output structure, the set points and partition of the fuzzy sets for the inputs and outputs were the same used in the previous method. The simulation results are shown in the Figure 14.17. The steady state error of blood pressure was reduced dramatically but the steady state error of muscle relaxation still exists. Hence, the SOFLC algorithm can be utilized to further fine-tune the rule-bases for each case according to previous experience.

14.4.4 Self-Organizing Fuzzy Logic Control of Four-Input/Two-Output Anesthesia System

The SOFLC system was applied to multivariable structure of four-input / two-output anesthesia simulation system as shown in Figure 14.18. In order to



(a)



(b)

Fig. 14.20. The rule-bases of four-input and two-output for SOFLC (a) Atracurium rule-base (b) Isoflurane rule-base (Notation: The italic and underline rules in the atracurium and isoflurane rule-bases generated by SOFLC)

compare with the basic fuzzy logic controller, the set points and partition of fuzzy sets of the inputs and outputs were the same used in the previous method. Also, in this simulation, the initial rule-base (i.e., 25 rules) that is generated for the simple FLC was used. The simulation results are shown in the Figure 14.19. The steady state error of muscle relaxation was reduced dramatically and the steady state error of blood pressure is better controlled due to the addition of new rules which were generated by self-organizing fuzzy logic structure as shown in Figure 14.20.

Finally, the steady state errors of these four methods were are tabulated in Table 14.1. It is shown that the steady state error of the SOFLC with four-input / two-output structure is the smallest in terms of muscle relaxation and blood pressure compared to that of the other three multivariable structures.

14.5 Conclusions

In this chapter, we have demonstrated that a multivariable SOFLC can provide more stable muscle relaxation and blood pressure by administering atracurium infusion rate and isoflurane concentration when rule-base modifications have been considered in comparison with a simple fuzzy logic which has fixed rule-bases. Two important aspects have been addressed in this chapter for simulating anesthesia control in the operating theatre. First, using decomposition of multivariable self-organizing fuzzy logic structure, we are able to handle the performance index and rule-base in multidimensional space. Second, the SOFLC algorithm has a learning ability which is similar to the way in which human experts use experiential knowledge or no knowledge to learn a clinical rule-base protocol for anesthesia control (i.e., learning-by-example). In this chapter, several new rules, which are not in the initial rule-base, were generated by the self-organizing learning process via 150 min simulation. Moreover, the simulations explored how the multivariable SOFLC algorithm compensates for the missing knowledge from the initial rule-bases, this evidence provides an insight view on how rules migrate and converge.

However, this study demonstrates the feasibility and applicability of the multivariable SOFLC in anesthesia control. But, it still needs a series of clinical trials at operating theatre, perhaps to refine the multivariable SOFLC, and certainly to show how widely they can be applied. Therefore, this presentation is by no means complete and it aims to give an idea of whether multivariable SOFLC can mimic human being thinking for monitoring multiple sensors and administering multiple drugs in the operating theatre. Also it can show whether the decomposition of multivariable self-organizing fuzzy logic structure can provide better performance when the rule-bases are modified. In this sense, an initial rule-base (i.e., 25 rules) is from simple decomposed fuzzy logic controllers that construct a 2-input / 2-output or 4-input / 2-output structure. The SOFLC features will modify the decomposed rule-bases separately to give the multivariable algorithm strength in combating complex systems. The current research can now be expanded to encompass alternative intravenous techniques and

different intravenous drugs (e.g., propofol, midazolam, morphine). In addition, the multivariable SOFLC could be expanded to include other closed-loop control in analgesia or NICU, such as multivariable pain control in extra corporeal shock wave lithotripsy [30] and even more complex multivariable control problems, such as the treatment of cerebral perfusion for controlling MAP and ICP in NICU [10].

Currently, fuzzy logic, neural networks and genetic algorithms are three popular artificial intelligence techniques that are widely used in many applications. Due to their distinct properties and advantages, they are currently being investigated and integrated to form models or strategies in the areas of system control. In control engineering, the fusion of fuzzy logic, neural networks and genetic algorithms is steadily growing [2, 33]. Therefore, using the hybrid intelligent approach to auto-tuning the parameters of the fuzzy logic controller may provide more suitable clinical control of anesthesia in operating theatre. However, the characteristics of on-line self-learning have lead the SOFLC to be more suitable for real time control in comparison with off-line analysis of the neural networks and genetic algorithms which are more time consuming.

References

1. Adams, A.P., Cashman, J.N.: Anaesthesia, analgesia and intensive care. Edward Arnold, London (1991)
2. Allen, R., Smith, D.: Neuro-fuzzy closed-loop control of depth of anaesthesia. Artificial Intelligence in Medicine 21, 185–191 (2001)
3. Chuang, C.T., Fan, S.Z., Shieh, J.S.: Muscle relaxation controlled by automated administration of cisatracurium. Biomedical Engineering- Applications, Basis & Communications 18(6), 284–295 (2006)
4. Chuang, C.T., Fan, S.Z., Shieh, J.S.: The use of intensive manual control to model cisatracurium pharmacokinetics and pharmacodynamics for neuromuscular block. Journal of Medical and Biological Engineering 26(4), 187–193 (2006)
5. Ciresi, G., Akay, M.: Fuzzy logic in medical control applications. Biomedical Engineering-Applications Basis Communications 8(6), 471–487 (1996)
6. Elkfafi, M., Shieh, J.S., Linkens, D.A., Peacock, J.E.: Intelligent signal processing of evoked potentials for anaesthesia monitoring and control. IEE Proc. Control Theory and Appl. 144(4), 354–360 (1997)
7. Elkfafi, M., Shieh, J.S., Linkens, D.A., Peacock, J.E.: Fuzzy logic for auditory evoked response monitoring and control of depth of anaesthesia. Fuzzy Sets and Systems 100, 29–43 (1998)
8. Gao, Y., Er, M.J.: Online adaptive fuzzy neural identification and control of a class of mimo nonlinear systems. IEEE Transactions on Fuzzy Systems 11(4), 462–477 (2003)
9. Gupta, M.M., Kiszka, J.B., Trojan, G.M.: Multivariable structure of fuzzy control systems. IEEE Transactions on Systems, Man, and Cybernetics 16(5), 638–656 (1986)
10. Huang, S.J., Shieh, Fu, M., Kao, M.C.: Fuzzy logic control for intracranial pressure via continuous propofol sedation in a neurosurgical intensive care unit. Medical Engineering & Physics 28(7), 639–647 (2006)

11. Kim, Y.T., Bien, Z.: Robust self-learning fuzzy controller design for a class of nonlinear mimo systems. Fuzzy Sets and Systems 111(2), 117–135 (2000)
12. Kosko, B.: Neural networks and fuzzy systems. Prentice-Hall International, Inc., Singapore (1991)
13. Linkens, D.A.: Adaptive and intelligent control in anesthesia. IEEE Control Systems Magazine, 6–11 (1992)
14. Linkens, D.A.: The role of intelligent systems engineering in biomedicine. Biomedical Engineering - Applications Basis Communications 8(5), 385–391 (1996)
15. Linkens, D.A., Hasnain, S.B.: Self-organizing fuzzy logic control and application to muscle relaxant anaesthesia. IEE Proceedings, Part D 138, 274–284 (1991)
16. MacLeod, A.D., Asbury, A.J., Gray, W.M., Linkens, D.A.: Automatic control of neuromuscular block with atracurium. British Journal of Anaesthesia 63, 31–35 (1989)
17. Mahfouf, M.: Generalised predictive control (gpc) in the operating theatre. In: Linkens, D.A. (ed.) Intelligent Control in Biomedicine, pp. 37–78. Taylor & Francis, London (1994)
18. Mahfouf, M., Abbod, M.F.: A comparative study of generalized predictive control (gpc) and intelligent self-organizing fuzzy logic control (soflc) for multivariable anaesthesia. In: Linkens, D.A. (ed.) Intelligent Control in Biomedicine, pp. 79–132. Taylor & Francis, London (1994)
19. Mahfouf, M., Abbod, M.F., Linkens, D.A.: Survey of fuzzy logic monitoring and control utilization in medicine. Artificial Intelligence in Medicine 21(1), 27–42 (2001)
20. Mahfouf, M., Linkens, D.A., Asbury, A.J., Gray, W.M., Peacock, J.E.: Generalised predictive control (gpc) in the operating theatre. IEE Proceedings, Part D 139, 404–420 (1992)
21. Martin, J.F., Smith, N.T., Quinn, M.L., Schneider, A.M.: Supervisory adaptive control of arterial pressure during cardiac surgery. IEEE Transactions on Biomedical Engineering 39(4), 389–393 (1992)
22. Mason, D.G., Ross, J.J., Edwards, N.D., Linkens, D.A., Reilly, C.S.: Self-learning fuzzy control of atracurium-induced neuromuscular block during surgery. Medical & Biological Engineering & Computing 35, 498–503 (1997)
23. Meier, R., Nieuwland, J., Zbinden, A.M., Hacisalihzade, S.S.: Fuzzy logic control of blood pressure during anesthesia. IEEE Control Systems Magazine 12(12), 12–17 (1992)
24. Millard, R.K., Monk, C.R., Woodcock, T.E., Roberts, C.P.: Controlled hypotension during ent surgery using self-tuners. Computational Biology and Medicine 17, 1–18 (1988)
25. Nie, J., Lee, T.H.: Self-organizing rule-based control of multivariable nonlinear servomechanisms. Fuzzy Sets and Systems 3, 285–304 (1997)
26. Procyk, T.J., Mamdani, E.H.: A linguistic self-organizing process controller. Automatica 15, 15–30 (1979)
27. Ross, J.J., Mason, D.G., Linkens, D.A., Edwards, N.D.: Self-learning fuzzy logic control of neuromuscular block. British Journal of Anesthesia 78, 412–415 (1997)
28. Shieh, J.S., Chang, L.W., Fan, S.Z., Liu, C.C.: Fuzzy logic control of propofol infusion using quantitative and qualitative approaches. Biomedical Engineering-Applications, Basis & Communications 9(6), 350–360 (1997)
29. Shieh, J.S., Chang, L.W., Fan, S.Z., Liu, C.C., Huang, H.H.: Automatic control of anaesthesia using hierarchical structure. Biomedical Engineering-Applications, Basis & Communication 10(4), 195–202 (1998)

30. Shieh, J.S., Chang, L.W., Yang, T.C., Liu, C.C.: An enhanced patient controlled analgesia (epca) for the extracorporeal shock wave lithotripsy (eswl). Biomedical Engineering-Applications, Basis & Communications 19(1), 7–17 (2007)
31. Shieh, J.S., Fan, S.Z., Chang, L.W., Liu, C.C.: Hierarchical rule-based monitoring and fuzzy logic control for neuromuscular block. Journal of Clinical Monitoring and Computing 16, 583–592 (2000)
32. Shieh, J.S., Fan, S.Z., Shi, W.L.: The intelligent architecture for simulation of inhalational anaesthesia. Biomedical Engineering-Application, Basis & Communications 16(5), 272–280 (2004)
33. Shieh, J.S., Kao, M.H., Liu, C.C.: Genetic fuzzy modelling and control of bispectral index (bis) for general intravenous anaesthesia. Medical Engineering & Physics 28(2), 134–148 (2006)
34. Shieh, J.S., Linkens, D.A., Asbury, A.J.: A hierarchical system of on-line advisory for monitoring and controlling the depth of anesthesia using self-organizing fuzzy logic. Engineering Applications of Artificial Intelligence 18(3), 307–316 (2005)
35. Shieh, J.S., Linkens, D.A., Peacock, J.E.: Hierarchical rule-based and self-organizing fuzzy logic control of anesthesia. IEEE Transaction on Systems, Man, and Cybernetics, Part C: Applications and Reviews 9(1), 98–109 (1999)
36. Shieh, J.S., Linkens, D.A., Peacock, J.E.: A computer screen-based simulator for hierarchical fuzzy logic monitoring and control of depth of anaesthesia. Mathematics and Computers in Simulation 67(3), 251–265 (2004)
37. Tatnall, M.L., Morris, P., West, P.G.: Controlled anaesthesia: An approach using patient characteristics identified during uptake. British Journal of Anaesthesia 53, 1019–1026 (1981)
38. Tong, R.M.: Synthesis of fuzzy models for industrial processes – some recent results. International Journal of General Systems 4, 143–162 (1978)
39. Uys, P.C., Morrell, D.F., Bradlow, H.S., Rametti, L.B.: Self-tuning microprocessor-based closed-loop control of atracurium-induced neuromuscular blockade. British Journal of Anaesthesia 61, 685–692 (1988)
40. Zbinden, A.M., Feigenwinter, P., Petersen-Felix, S., Hacisalihzade, S.: Arterial pressure control with isoflurane using fuzzy logic. British Journal of Anaesthesia 74, 66–72 (1995)

15

Interval Type-2 Fuzzy System for ECG Arrhythmic Classification

Teck Wee Chua and Woei Wan Tan

Department of Electrical Engineering, National University of Singapore, Singapore
{cteckwee,wwtan}@nus.edu.sg

Summary. This chapter presents an Interval Type-2 fuzzy classifier and its application to ECG arrhythmic classification problem. The uncertainties associated with the membership functions are encapsulated by the footprint of uncertainty (FOU) and it is totally characterized by the upper membership function (UMF) and lower membership function (LMF). To enable designed membership functions (MFs) reflect the data, we proposed three types of FOU design strategies according to the dispersion of the data. The first and second designs comprise of Gaussian MFs with uncertain standard deviations and means respectively whereas the third design is the combination of both. The FOU is then further optimized through Genetic Algorithm. The proposed Type-2 fuzzy classifier has been applied to ECG arrhythmic classification problem to discriminate three types of ECG signals, namely the normal sinus rhythm (NSR), ventricular fibrillation (VF), and ventricular tachycardia (VT). The performance of the classifier is tested on MIT-BIH Arrhythmia database. The average period and pulse width of ECG data are extracted as the inputs to the classifier. Different sources of noises have been included to model the uncertainties associated with the vagueness in MFs and the unpredictability of the data. The results show that the proposed strategies to design the FOU are essential to achieve a high performance fuzzy rule-based classifier in face of the uncertainties.

15.1 Introduction

Fuzzy systems have been used successfully in an increasing number of application areas. One of the main advantages of fuzzy logic is that it enables qualitative domain knowledge about a classification task to be deployed in the algorithmic structure. Fuzzy approaches to pattern recognition have been pioneered by Bellman, Kalaba and Zadeh [1]. In recent years, it has been successfully applied in medical domains. There are many inherent virtues of fuzzy system which are suitable for medical applications. Firstly, it can avoid hard threshold, thereby increasing the tolerance towards contradictions in the data. Secondly, a fuzzy rule-based classifier (FRBC) provides a framework to incorporate both subjective (i.e., expert opinion) and objective (i.e., design samples where the knowledge can be extracted) information, hence it may be able to outperform other classifiers. It is possible to integrate this valuable knowledge into the fuzzy logic system due to the system's similar reasoning style to the human being. Thirdly, biomedical features have statistical attributes that are non-stationary and mathematical descriptions of the non-stationarities are unknown [6].

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 297–314.
springerlink.com

Different types of fuzzy systems may cater for different sources of uncertainties. Mendel [6] suggests that a non-singleton type-1 fuzzy logic system (FLS) can be used to handle uncertainties caused by noisy measurements. The inputs are modeled as type-1 fuzzy numbers. A mathematical analysis [2] shows that the non-singleton fuzzifier manages to minimize the effect of noise. Despite the popularity of fuzzy logic, an ordinary (type-1) fuzzy set does not capture uncertainty in all of its manifestations, particularly when it arises from vagueness in the shape of the membership function. The uncertainties of the MFs could arise from differing expert opinions which are used to formulate the fuzzy rules or due to the noisy inputs when they are used to train the FLS. As such, the imprecise boundaries of a type-2 fuzzy set give rise to truth or membership values that are fuzzy instead of a crisp number, may overcome the problem. In particular, the MF of a type-2 fuzzy set has blur boundary and consists of a set of admissible type-1 MFs. The employment of general type-2 FLS usually increases the computational complexity in comparison with type-1 FLS, therefore the simplified version of general type-2 FLS, which is known as interval Type-2 FLS, is preferable. The reduction on the computational complexity is due to the property that all the secondary membership grades for an interval type-2 fuzzy set are all equal to one rather than a value in [0,1].

In this chapter, we investigate, through the application of ECG arrhythmic classification, the feasibility of capturing the uncertainties associated with the antecedent sets and the inputs via the type-2 fuzzy framework. Moreover, we also show how it is straightforward to directly exploit the information inherent in the problem to set up the number of fuzzy rules, and the antecedent set parameters. This chapter is organized as follows. Section 15.2 provides fundamental theory about an interval type-2 fuzzy set [7]. The classification problem and feature extraction method are explained in Section 15.3. Section 15.4 outlines the interval type-2 fuzzy rule-based classifier (FRBC) and the proposed design methods. The experimental results are presented in Section 15.5 and finally Section 15.6 offers concluding remarks.

15.2 Interval Type-2 Fuzzy Set

An interval type-2 (IT2) fuzzy set, $\widetilde{A}$ is characterized as:

$$\widetilde{A} = \int_{x \in X} \int_{u \in J_x \subseteq [0,1]} 1/(x,u) = \int_{x \in X} \left[\int_{u \in J_x \subseteq [0,1]} 1/u \right] /x \tag{15.1}$$

where x, the primary variable, has domain X; $u \in U$, the secondary variable, has domain J_x at each $x \in X$; J_x is called the primary membership of x and is defined in (15.5); and, the secondary grades of $\widetilde{A}$ all equal 1. Note that (15.1) means: $\widetilde{A} : X \rightarrow \{[a,b] : 0 \leq a \leq b \leq 1\}$. Uncertainty about the shape and position of $\widetilde{A}$ is conveyed by the union of all the primary memberships, which is called the footprint of uncertainty (FOU) of $\widetilde{A}$ (see Fig. 15.1), i.e.

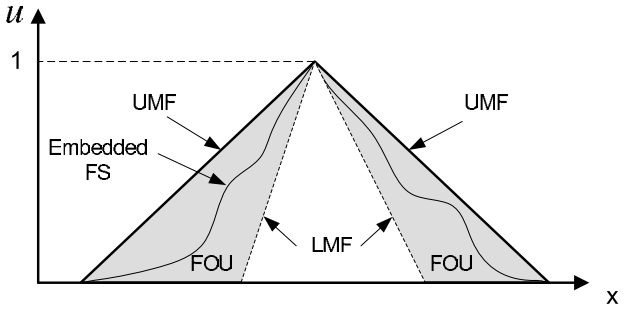


Fig. 15.1. FOU (shaded), LMF (dashed), UMF (solid) and an embedded FS (wavy line) for IT2 FS $\widetilde{A}$

$$FOU(\widetilde{A}) = \bigcup_{\forall x \in X} J_x = \{(x,u) : u \in J_x \subseteq [0,1]\} \tag{15.2}$$

The upper membership function (UMF) and lower membership function (LMF) of $\widetilde{A}$ are two type-1 MFs that bound the FOU (Fig. 15.1). The UMF is associated with the upper bound of FOU and is denoted $\bar{\mu}_{\widetilde{A}}$, $\forall x \in X$, and the LMF is associated with the lower bound of FOU and is denoted $\underline{\mu}_{\widetilde{A}}$, $\forall x \in X$, i.e.

$$\bar{\mu}_{\widetilde{A}}(x) \equiv \overline{FOU(\widetilde{A})} \quad \forall x \in X \tag{15.3}$$

$$\underline{\mu}_{\widetilde{A}}(x) \equiv \underline{FOU(\widetilde{A})} \quad \forall x \in X \tag{15.4}$$

Note that J_x is an interval set, i.e.

$$J_x = \{(x,u) : u \in [\underline{\mu}_{\widetilde{A}}(x), \bar{\mu}_{\widetilde{A}}(x)]\} \tag{15.5}$$

so that $FOU(\widetilde{A})$ in (15.2) can also be expressed as

$$FOU(\widetilde{A}) = \bigcup_{\forall x \in X} [\underline{\mu}_{\widetilde{A}}(x), \bar{\mu}_{\widetilde{A}}(x)]. \tag{15.6}$$

For continuous universes of discourse X and U, an embedded IT2 FS $\widetilde{A}_e$ is

$$\widetilde{A}_e = \int_{x \in X} [1/u]/x \quad u \in J_x \tag{15.7}$$

Note that (15.7) means: $\widetilde{A}_e : X \rightarrow \{u : 0 \leq u \leq 1\}$. The set $\widetilde{A}_e$ is embedded in $\widetilde{A}$ such that at each x it only has one secondary variable (i.e., one primary membership whose secondary grade equals 1). Examples of $\widetilde{A}_e$ are[1] $\bar{\mu}_{\widetilde{A}}(x)$ and $\underline{\mu}_{\widetilde{A}}(x)$, $\forall x \in X$.

[1] In this notation it is understood that the secondary grade equals 1 at all elements in $\bar{\mu}_{\widetilde{A}}(x)$ or $\underline{\mu}_{\widetilde{A}}(x)$.

For discrete universes of discourse X and U, in which x has been discretized into N values and at each of these values u has been discretized into M_i values, an embedded IT2 FS $\widetilde{A}_e$ has N elements, where $\widetilde{A}_e$ contains exactly one element from $J_{x_1}, J_{x_2}, \ldots, J_{x_N}$, namely $u_1, u_2, \ldots, u_N$, each with a secondary grade equal to 1, i.e., $\widetilde{A}_e = \sum_{i=1}^{N} [1/u_i]/x_i$, where $u_i \in J_{x_i}$. Set $\widetilde{A}_e$ is embedded in $\widetilde{A}$, and, there are a total of $\prod_{i=1}^{N} M_i$ $\widetilde{A}_e$. Associated with each $\widetilde{A}_e$ is an embedded T1 FS A_e, where

$$A_e = \int_{x \in X} u/x \quad u \in J_x \tag{15.8}$$

Note that (15.8) means $A_e : X \rightarrow \{u : 0 \leq u \leq 1\}$. The set A_e, which acts as the domain for $\widetilde{A}_e$, is the union of all the primary memberships of the set $\widetilde{A}_e$ in (15.7). As the universes of discourse X and U are continuous then there is an uncountable number of embedded IT2 FSs ($\widetilde{A}_e$) and embedded T1 FSs (A_e) in $\widetilde{A}$. Because such sets are only used for theoretical purposes and are not used for computational purposes, this poses no problem. For discrete universes of discourse X and U, an embedded T1 FS A_e has N elements, one each from $J_{x_1}, J_{x_2}, \ldots, J_{x_N}$, namely $u_1, u_2, \ldots, u_N$, i.e., $A_e = \sum_{i=1}^{N} [u_i]/x_i$ where $u_i \in J_{x_i}$. Set A_e is the union of all the primary memberships of $\widetilde{A}_e$ and there are a total of $\prod_{i=1}^{N} M_i$ A_e. For simplicity, without explicitly stated, we assume that the term type-2 fuzzy set is referring to interval type-2 fuzzy set in the remainder of this chapter.

15.3 Problem Description

An electrocardiogram (ECG) is the representation of the electrical activity of the heart (cardiac) muscle as it is recorded from the body surface. Fig. 15.2 shows the various components of a typical ECG signal. The P wave represents depolarization of the upper part of the heart, the atria whereas the QRS complex represents ventricular depolarization and T wave represents ventricular repolarization.

Changes in normal rhythm of a human heart may result in different cardiac arrhythmias, which may be immediately fatal or cause irreparable damage to the heart when sustained over long periods of time. Ventricular fibrillation (VF) and ventricular tachycardia (VT) are both life-threatening cardiac arrhythmias.

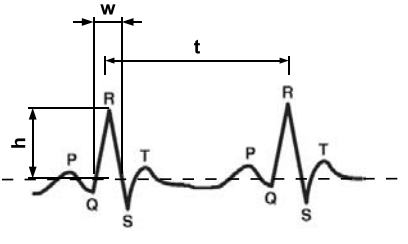


Fig. 15.2. ECG components: P wave, QRS complex, and T wave

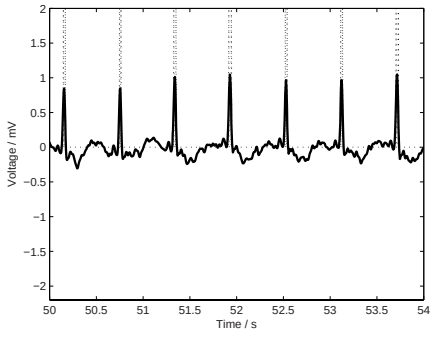
In particular, VF requires immediate defibrillation whereas VT must be distinguished from Normal Sinus Rhythm (NSR) and VF to receive cardioversion, by delivering a shock of somewhat lower energy in synchronization with the heart beat. Critical cardiac incidents occur most often out of hospitals, therefore automatic external defibrillators (AED) were introduced for increasing the survival rate [3]. Since the successful termination of VF and VT requires fast response and application of high-energy shocks in the heart region, the accuracy of the built-in algorithm for VF detection is of paramount importance. One major influence on the detection accuracy is the capability of the classifier to account for the uncertainties.

15.3.1 Data and Feature Extraction

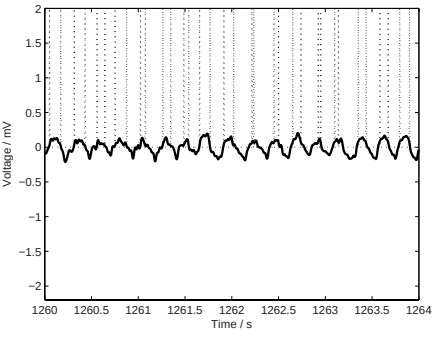
The ECG data used in this study is obtained from MIT-BIH Malignant Ventricular Arrhythmia Database (VFDB) [8]. All signals are first preprocessed by a 0.05-40Hz bandpass filter and a 60Hz notch filter in order to suppress DC components, baseline drifts and possible electrical interference. The filtered ECG signal are transformed into the binary strings. The transformation algorithm used in this chapter is a modified version of that in the paper [11]. Unlike Zhang's one-pass conversion, a two-pass conversion method is employed. The ECG signal will be transformed into partial binary string first instead of full binary string directly. This can reduce the false positive peak detection greatly by eliminating low amplitude signal. This step is closely followed by a full binary string conversion for determining a threshold that can maximize the differences between NSR class and VF/VT classes. The steps are listed as follows:

1. Select a finite length (i.e., 4s) of ECG. Since the VFDB signals were digitized at 250Hz, then there will be 1000 data points $\{x_i | i = 1, 2, \ldots, n;\ n = 1000\}$ within 4s window length.
2. Mean-center ECG data where the mean data, x_m is subtracted from every data point, i.e., $\{x_i - x_m\}$.
3. Find out the negative peak, V_n and positive peak V_p.
4. Form a partial binary string: if the signal level falls in between the range of $(0 < x_i < 0.2\ V_p)$ or $(0.2\ V_n < x_i < 0)$, then it is assigned "0".
5. Calculate the parameters N_p and N_n. N_p denotes the number of data $(x_i > 0)$ while $N_n = n - N_p$.
6. Determine a proper threshold, T_r to convert the partial binary string into a complete binary string: if $N_p < 0.15\ n$, then threshold is assigned as $T_r = 0.7\ V_p$, otherwise $T_r = 0$. This step is crucial to separate NSR signals from VF and VT signals.
7. Compare x_i to T_r to turn the partial binary string into a complete binary string, that is if $x_i \leq T_r$, then x_i is assigned as "0" or otherwise "1".

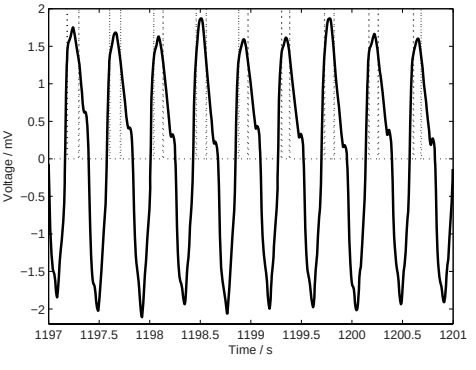
The graphs in Fig. 15.3 show the examples of three different types of ECG signals with their corresponding binary sequences. Two time-domain features commonly used for ECG classification are extracted from the binary sequences:



(a)



(b)



(c)

Fig. 15.3. ECG signals (excerpts from VFDB) and corresponding binary sequences: (a) NSR, record 421 (50-54s), (b) VF, record 424 (1260-1264s), (c) VF, record 611(1197-1201s)

pulse width, and *pulse period* and the scatter plot is shown in Fig. 15.4. All parameters are averaged within the 4s window. For the sake of convenience, all attribute values in this chapter were normalized into real number between unit interval $[0, 1]$ as:

$$x_{i,k} := \frac{x_{i,k} - \min\{x_{k|\forall i}\}}{\max\{x_{k|\forall i}\} - \min\{x_{k|\forall i}\}} \tag{15.9}$$

where $i = 1, \ldots, N,\ k = 1, \ldots, p$. Therefore, the C-class classification problem is defined in the p-dimensional unit cube $[0, 1]^p$.

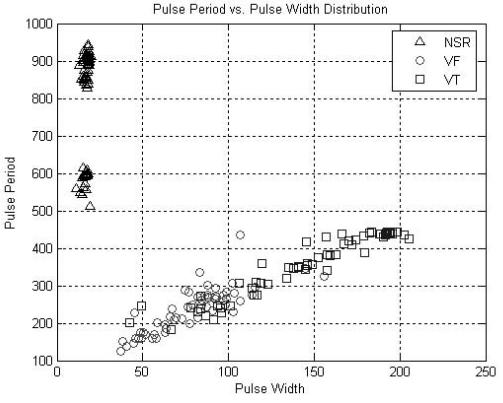


Fig. 15.4. The scatter plot for inputs pulse period vs. width

15.4 Fuzzy Rule-Based Classifiers (FRBCs)

There are two popular types of FRBCs: Mamdani-Assilian (MA) type and Takagi-Sugeno-Kang (TSK) type. The MA classifier requires both the input and output domains to be characterized by linguistic terms. Both the antecedent and consequent of an if-then rule are typically Boolean expressions of simple clauses. The TSK type has the same Boolean expressions of simple clauses for the antecedent part. However, the consequent is a function of the input (e.g. a polynomial). In this chapter, we only focus on MA type FRBC.

15.4.1 Interval Type-2 Fuzzy Rule-Based Classifier Structure

This sub-section introduces the interval type-2 FRBC. Fig. 15.5 shows the general structure of the proposed type-2 fuzzy rule-based classifier. There are six components in the architecture. The rule-base consists of T rules where each rule relates the domain $X_1 \times \cdots \times X_p \subseteq R^p$ to the range $Y \in R$ and can be expressed as the following intuitive IF-THEN statement:

$$R^j\text{: IF } x_1 \text{ is } \widetilde{A}_1^j \text{ and } \cdots x_p \text{ is } \widetilde{A}_p^j\text{, THEN } y \text{ is } C^j$$

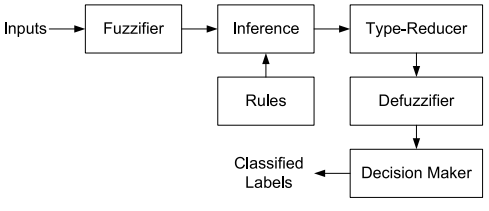


Fig. 15.5. Structure of type-2 classifier

where R^j denotes the jth rule, $\widetilde{A}_k^j$ is an interval type-2 antecedent set associated with the kth input variable x_k $(k = 1, \ldots, p)$, and C^j represents the consequent set associated with the output variable y. The role of the fuzzifier in a fuzzy system is to map each of the element, x'_k, in the input vector $\mathbf{x}' = (x'_1, \ldots, x'_p)^T$ into the fuzzy set $\widetilde{X}'_k$. This process provides a natural framework for handling uncertain input information. There is a variety of methods for performing fuzzification. The most common approach is singleton fuzzification, which maps a crisp input into the following MF:

$$\mu_{\widetilde{X}'_k}(x_k) = \begin{cases} 1 & x_i = x'_k \\ 0 & x_k \neq x'_k \end{cases}$$

Next, the inference engine component computes the firing strengths for each rule which expresses how well the the fuzzified input $\widetilde{X}'$ match the antecedents $\widetilde{A}'$. For type-2 FRBC, the inference engine produces two firing strengths for each rule, the lower and upper firing strengths of the jth rule, $\underline{f}^j(\mathbf{x}')$ and $\bar{f}^j(\mathbf{x}')$, are computed as:

$$\underline{f}^j(\mathbf{x}') = \prod_{k=1}^{p} \sup_{x_k} [\mu_{\widetilde{X}'_k}(x_k), \underline{\mu}_{\widetilde{A}_k^j}(x_k)] \tag{15.10}$$

$$\bar{f}^j(\mathbf{x}') = \prod_{k=1}^{p} \sup_{x_k} [\mu_{\widetilde{X}'_k}(x_k), \bar{\mu}_{\widetilde{A}_k^j}(x_k)] \tag{15.11}$$

where sup[.] denotes supremum operation. Before the final crisp output can be obtained, the output of the inference engine and the consequent must be processed. In a more general case where the consequent fuzzy sets $\widetilde{C}^j$ are interval type-2 sets, the type-reduced set Y_{cos} can be computed with center-of-sets type reduction:

$$Y_{cos} = [y_l, y_r] = \int_{y^1 \in [y_l^1, y_r^1]} \cdots \int_{y^M \in [y_l^M, y_r^M]} \int_{f^1 \in [\underline{f}^1, \bar{f}^1]} \cdots \int_{f^M \in [\underline{f}^M, \bar{f}^M]} 1 \Big/ \frac{\sum_{j=1}^{M} f^j y^j}{\sum_{j=1}^{M} f^j} \tag{15.12}$$

where $[y_l^j, y_r^j]$ denotes to the centroid of the set $\widetilde{C}^j$, which can be obtained from various methods defined in [5]. However, the consequent fuzzy sets in our classification problem correspond to the ECG arrhythmia labels and are represented by crisp number (singleton), the center-of-sets type-reduction above is simplified to height type-reduction by simply setting $y_l^j = y_r^j$. The type-reduced set which is an interval output, $[y_l(\mathbf{x}'), y_r(\mathbf{x}')]$ can be obtained via Karnik-Mendel iterative algorithm [7]. To compute y_l, the steps are:

1. Without loss of generality, assume that pre-computed y_r^j are arranged in ascending order; i.e., $y_r^1 \leq y_r^2 \leq \cdots \leq y_r^T$;
2. Compute y_r as $y_r = \sum_{j=1}^{T} f_r^j y_r^j / \sum_{j=1}^{T} f_r^j$ by initially setting $f_r^j = (\underline{f}^j + \bar{f}^j)/2$ for $j = 1, \ldots, T$ and let $y'_r \equiv y_r$;

3. Find R $(1 \leq R \leq T-1)$ such that $y_r^R \leq y_r' \leq y_r^{R+1}$;
4. Compute y_r as $y_r = \sum_{j=1}^{T} f_r^j y_r^j / \sum_{j=1}^{T} f_r^j$ with $f_r^j = \underline{f}^j$ for $i \leq R$ and $f_r^j = \bar{f}^j$ for $i > R$ and let $y_r'' \equiv y_r$;
5. If $y_r'' \neq y_r'$, then go to Step 6. If $y_r'' = y_r'$, then stop and set $y_r'' \equiv y_r$;
6. Set y_r' equal to y_r'', and return to Step 3.

The procedure for computing y_l is very similar to the one for y_r. Just replace y_r^j by y_l^j, and, in Step 3 find $L(1 \leq L \leq T-1)$ such that $y_l^L \leq y_l' \leq y_l^{L+1}$. Additionally, in Step 2 compute y_l as $y_l = \sum_{j=1}^{T} f_l^j y_l^j / \sum_{j=1}^{T} f_l^j$ by initially setting $f_l^j = (\underline{f}^j + \bar{f}^j)/2$ for $j = 1, \ldots, T$ and, in Step 4 compute y_l as $y_l = \sum_{j=1}^{T} f_l^j y_l^j / \sum_{j=1}^{T} f_l^j$ with $f_l^j = \bar{f}^j$ for $i \leq L$ and $f_l^j = \underline{f}^j$ for $i > L$.

The type-reduced set is then defuzzified to the crisp output, y by simply taking the average of y_l and y_r, i.e.:

$$y(\mathbf{x}') = \frac{y_l(\mathbf{x}') + y_r(\mathbf{x}')}{2} \tag{15.13}$$

Finally, the decision maker will determine the class label:

$$Class(\mathbf{x}') = \arg \min_j (y(\mathbf{x}') - \bar{C}^j) \tag{15.14}$$

where $\bar{C}^j$ denotes the singleton at the point having maximum membership in the jth consequent set. For Gaussian MF, this point is equal to the mean of the function.

15.4.2 Classifier Designs

In this sub-section, the design strategy of the type-2 classifiers will be explained. A useful trait of the design methodology is most of the antecedent MF parameters can be conveniently derived from data itself. The design strategy, which comprises four steps, is summarized in Fig. 15.6. The first step is to determine the structure of the classifier. This can be achieved by establishing one fuzzy rule for each naturally distinguishable class. Since there are three classes, it would be intuitive to form only three rules (i.e., $T = 3$) in this problem. In addition,

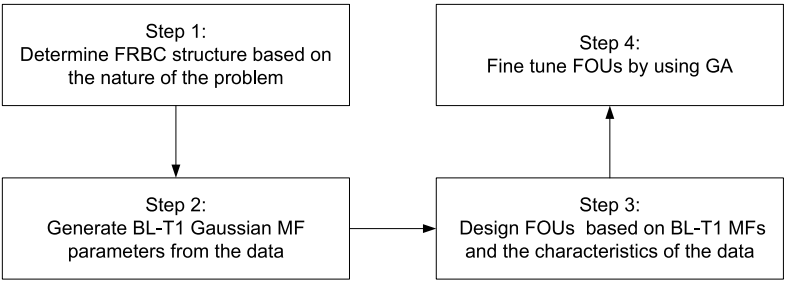


Fig. 15.6. The design strategy of Type-2 FRBCs

the number of antecedents for each rule is determined by the number of features which is two in our case. The next step is to determine the parameters of the MFs. To ensure the designed MF parameters are relevant to the data and to achieve good interpretability and transparency of the rule-base, our second strategy is to design the prototype type-1 MF where the mean M_k^j and standard deviation σ_k^j of the Gaussian MF parameter are computed according to the distribution of the data. For $\forall x \in Classj$:

$$M_k^j = \frac{1}{N_j} \sum_{i=1}^{N_j} x_{i,k} \tag{15.15}$$

$$\sigma_k^j = \sqrt{\sum_{i=1}^{N_j} \frac{1}{N_j} (x_{i,k} - M_k^j)^2} \tag{15.16}$$

where N_j denote the total number of samples from class j. Alternatively, the mean of type-1 MF can be computed with other more advanced clustering algorithms like Fuzzy C-Means (FCM), Self-Organizing Map (SOM) etc. We define these prototype MFs as base-line type-1 (BS-T1) MFs. Based on the BL-T1 MFs, we design three types of FOUs with the aim to account for different sources of uncertainties such as randomness of the data and the ambiguity in determining the exact membership functions. The first type-2 MF is shown in Fig. 7(a) where the upper membership function (UMF) is characterized by BL-T1 MF. The lower membership function (LMF) has the same mean as the UMF but with two different standard deviations, ($\underline{\sigma}_{L,k}^j$, $\underline{\sigma}_{R,k}^j$). The idea to incorporate two different standard deviations is motivated by the fact that most classification problems have uneven data distribution with respect to the mean. For example, the ECG data distributions in Fig. 15.4 have different densities. The initial values of both parameters are then computed as:

$$\underline{\sigma}_{L,k}^j = \sqrt{\sum_{x \in Classj} \frac{1}{N_j'} (x_{i,k} - M_k^j)^2} \quad \text{for } \forall x_{i,k} \leq M_k^j \tag{15.17}$$

$$\underline{\sigma}_{R,k}^j = \sqrt{\sum_{x \in Classj} \frac{1}{N_j''} (x_{i,k} - M_k^j)^2} \quad \text{for } \forall x_{i,k} > M_k^j \tag{15.18}$$

where N_j' and N_j'' represent the total number of samples from class j which satisfy the condition parts of (15.17) and (15.18) respectively. Since the asymmetrical FOUs are created by varying the standard deviations, this classifier is named as the type-2 uncertain standard deviations (T2-US) classifier. The second one is known as type-2 uncertain means (T2-UM) classifier. As the name suggests, the UMF has two mean values, [$\bar{M}_{L,k}^j$, $\bar{M}_{R,k}^j$]. For $\forall x \in Classj$:

$$\bar{M}_{L,k}^j = \frac{1}{N_j'} \sum_{i=1}^{N_j'} x_{i,k} \quad \text{for } \forall x_{i,k} \leq M_k^j \tag{15.19}$$

$$\bar{M}_{R,k}^{j} = \frac{1}{N_j''} \sum_{i=1}^{N_j''} x_{i,k} \quad \text{for } \forall x_{i,k} > M_k^j. \tag{15.20}$$

This strategy is motivated by the limitations of clustering algorithms to locate the true mean of the MF. For example, the FCM algorithm is known to work well with evenly distributed data that are spherical in shape [4] but may not work well in the case of elliptical distribution. Fig. 15.4 clearly shows that NSR data comprise of two sub-clusters with high densities compared to VF and VT classes with sparse densities. Furthermore, all three classes have elliptical data distribution. The BL-T1 MF is used as the LMF for T2-UM classifier (see Fig. 7(b)). Finally, the third classifier- type-2 uncertain standard deviations and means (T2-USUM), is the combination of T2-US and T2-UM classifiers. We see that BL-T1 MF automatically served as the principal MF, as shown in Fig. 7(c). For all three types of type-2 classifiers, the steps above intend to capture as much uncertainty as possible from the data through the FOUs. The initial parameters such as LMF of T2-US classifier, UMF of T2-UM classifier and both LMF and UMF of T2-USUM classifier served as the good initial search points in the later stage of optimization by Genetic Algorithm (GA).

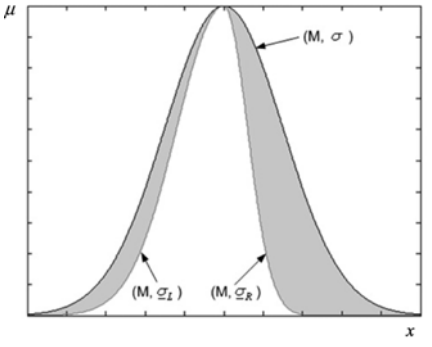
A GA is a search technique used in computing to find exact or approximate solutions to optimization and search problems. It is categorized as global search heuristics. A GA starts off with a population of randomly generated chromosomes, and advances toward better chromosomes by applying genetic operators. The population undergoes evolution in a form of natural selection. During successive iterations, called generations, chromosomes in the population are rated for their adaptation as solutions, and on the basis of these evaluations, a new population of chromosomes is formed using a selection mechanism and specific genetic operators (mutation and crossover). A fitness function is used to return a single numerical fitness of the individual in the population, which is supposed to be proportional to the utility or adaptation of the solution represented by that chromosome. Binary coded GA is adopted in the current framework. Each of the parameter is encoded in a 8-bit string. The training accuracy is chosen as the fitness function. During the fitness evaluation, the parameters are decoded into real numbers using linear mapping equation as shown below:

$$g_p = G_q^{min} + (G_q^{max} - G_q^{min}) \times \frac{A_q}{2^N - 1} \tag{15.21}$$

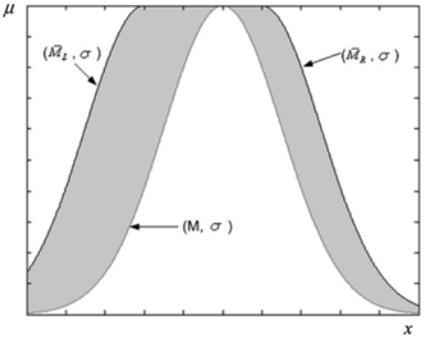
where g_p denotes the actual value of the q^{th} parameter, A_q denotes the integer represented by a N-bit string gene, G_q^{max} and G_q^{min} denote the user defined upper and lower limits of the gene respectively. The selection method is tournament size of two with elitism. As for the genetic operators, bitwise flipping mutation and single-point crossover are implemented.

15.5 Experimental Results

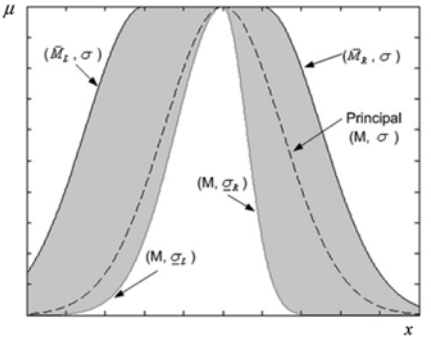
In this section, we will carry out five case studies to examine the performances of different types of T2 FRBCs. In addition, T2 FRBCs are compared against



(a)



(b)



(c)

Fig. 15.7. Interval type-2 Gaussian membership functions with: (a) uncertain standard deviations, (b) uncertain means, (c) uncertain standard deviations and means

BL-T1 FRBC. The structure of a BL-T1 FRBC is similar to a T2 FRBC (refer to Section 15.4.1) except for a few aspects. Firstly, the inference engine will produce a firing strength, f^j for jth rule rather than an interval value. Secondly, the type-reducer does not exist since no type-2 number is involved. For height defuzzification, the crisp output, y can be computed as:

$$y(\mathbf{x}') = \frac{\sum_{j=1}^{T} y^j f^j}{\sum_{j=1}^{T} f^j} \tag{15.22}$$

If all FOUs of a type-2 FRBC disappear, then type-2 FRBC is immediately reduced to type-1 FRBC and there is no difference between the final outputs from both classifiers.

The first case study seeks to examine the importance of the evolved FOUs in T2 FRBCs compared to the BL-T1 FRBC when the means of the MFs are obtained via Equation (15.15). The second case study focuses on whether the more advanced clustering algorithm can further improve the performance of T2 FRBCs. This is achieved by computing the means through the FCM algorithm. Subsequently, the third and fourth case studies examine the performances of T2-FRBCs when the evolved antecedent membership functions in case study 1 are later perturbed with noises. In the third case study, only the means of the MFs (as computed in first case study) are corrupted with noises while the fourth case study is configured in such a way that only the standard deviations of the MFs are perturbed with noises. In practice, the uncertainties could be due to different experts' opinions which are used to construct the rule-base. As suggested by Mendel [5], words can mean different things to different people. Therefore, there exists vagueness in the linguistic labels. Moreover, the random disturbances could be due to the noisy training data. The dynamics of the ECG signal is inherently noisy because it is very sensitive to cable movement and

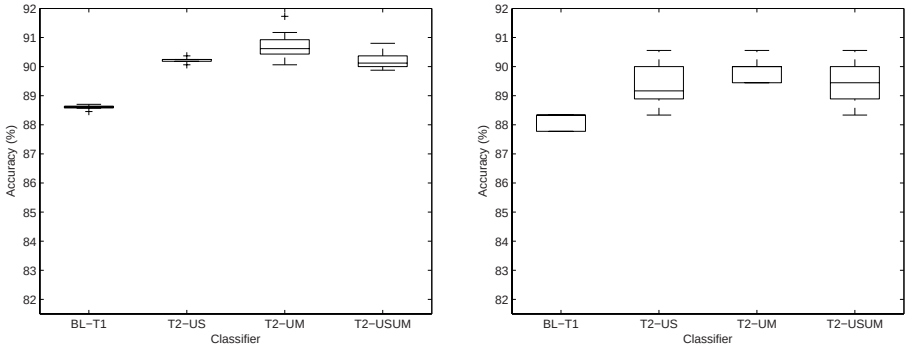


Fig. 15.8. Boxplot for case study 1 with 10-CV and ten iterations (a) training accuracy, (b) testing accuracy

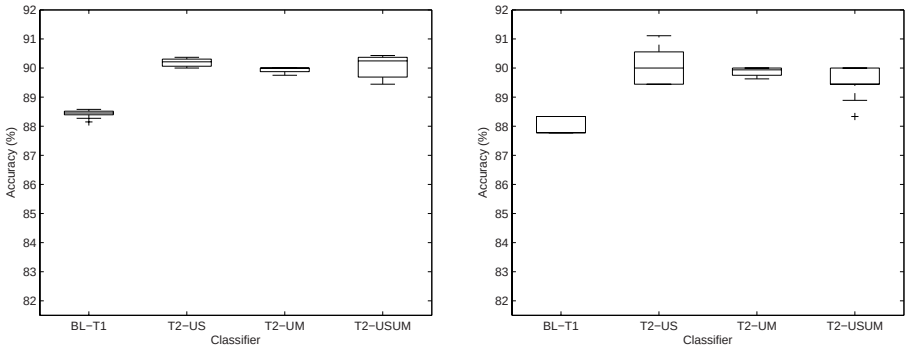


Fig. 15.9. Boxplot for case study 2 with 10-CV and ten iterations (a) training accuracy, (b) testing accuracy

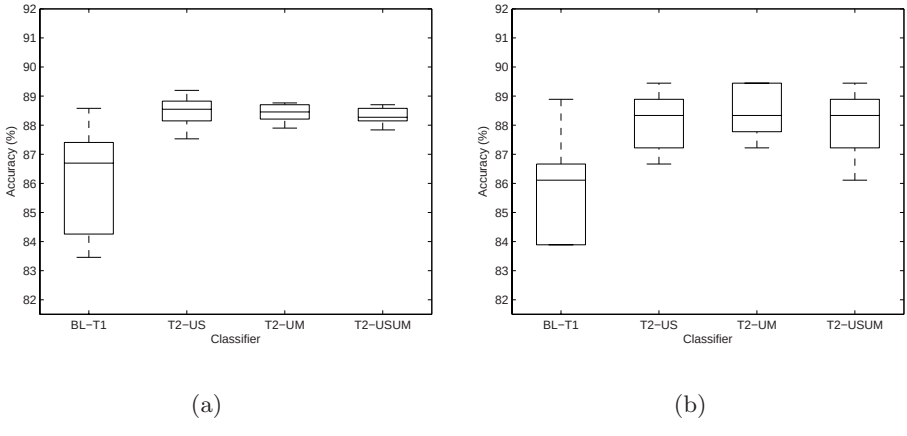
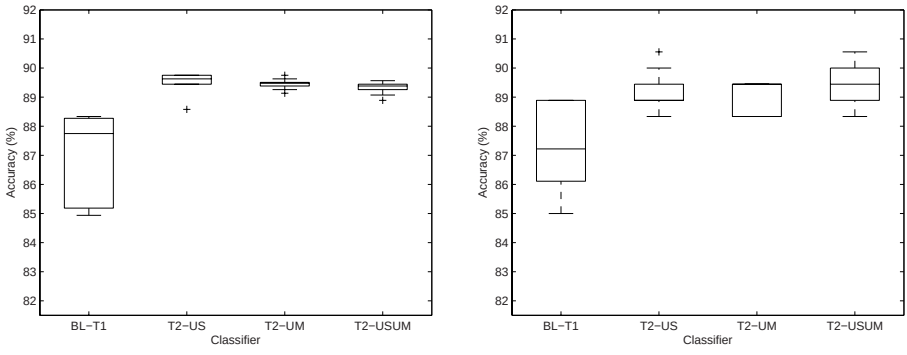


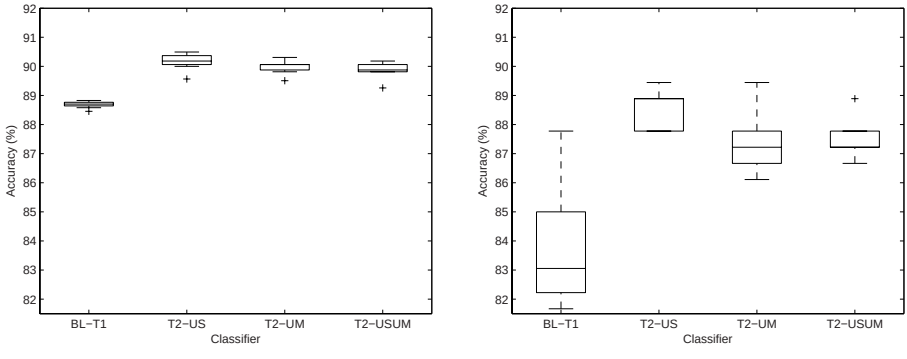
Fig. 15.10. Boxplot for case study 3 with 10-CV and ten iterations (a) training accuracy, (b) testing accuracy

muscle activity. In addition, the interference from electrical network can degrade the recording process especially for surface ECG recording. As the T2 FRBCs attempt to the encapsulate the aforementioned uncertainties in the FOUs of the antecedent sets, it may be able to outperform its T1 counterpart. The last case study aims to study if T2 FRBCs can handle the uncertainty associated with unpredictability [10], [9] better. Roughly speaking, this is one of the most important issues in ECG classification or any pattern classification problems. It reflects the situation where the applied testing samples deviate from the training samples to some extent. When referring to automated external defibrillator (AED), this occurs when the ECG signals produced by the patient deviate from



(a) (b)

Fig. 15.11. Boxplot for case study 4 with 10-CV and ten iterations (a) training accuracy, (b) testing accuracy



(a) (b)

Fig. 15.12. Boxplot for case study 5 with 10-CV and ten iterations (a) training accuracy, (b) testing accuracy

the training samples during the algorithm development stage. Hopefully, T2 FRBCs can improve the tolerance towards the unpredictability. This case study is modeled by noise corrupted test data while the training data are unperturbed. In case studies 3, 4 and 5, the noise source is modeled as a Gaussian function with zero mean and variance of 0.001:

$$P' = P + \sqrt{0.001} \times uniform() \tag{15.23}$$

where P represents either Gaussian MF mean or standard deviation parameter in case 3 and 4 or the input value in case 5 while P' represents the noise corrupted

Table 15.1. Average Training Accuracies of FRBCs (in %)

	BL-T1	T2-US	T2-UM	T2-USUM
Case 1	88.5926	90.2407	90.7407	90.2161
Case 2	88.4259	90.1914	89.9444	90.0432
Case 3	86.1729	88.4444	88.4259	88.3210
Case 4	87.0062	89.5309	89.4568	89.3148
Case 5	88.6914	90.1729	89.9876	89.8272

Table 15.2. Average Testing Accuracies of FRBCs (in %)

	BL-T1	T2-US	T2-UM	T2-USUM
Case 1	88.1111	89.3333	89.9444	89.3333
Case 2	88.0000	90.1111	89.8765	89.4444
Case 3	85.8334	88.1667	88.4444	88.0555
Case 4	87.1667	89.1667	89.0555	89.4444
Case 5	83.6667	88.6111	87.4444	87.5000

parameter. $uniform()$ denotes a function that generates a scalar value from a normal distribution with mean 0 and standard deviation 1.

To test the performance of each classifier, we carried out a 10-fold cross-validation (10-CV) procedure. For GA, we did not use adaptive parameters in order to keep the algorithm as simple as possible. The mutation rate was initially set to 0.1, 0.05, 0.03 and 0.01 respectively. It was noticed that mutation rates of 0.1 and 0.05 can lead to premature convergence occasionally. On the other hand, the convergence speed of the solution can be very slow when the mutation rate was set to 0.01. It worked out that mutation rate of 0.03 gave the best compromise between the convergence speed and classifier's accuracy. Likewise, we compared the crossover rates (0.5, 0.7, 0.8 and 0.9) and 0.8 consistently gave the best GA performance. The population size was 30 and the maximum number of generations was fixed at 100. The optimization process stops if there is no improvement in the fitness functions of the past 30 generations. In this particular application, the solutions usually converged between 70 and 90 generations. The results of the 10-CV experiments are summarized in Tables 15.1 and 15.2 where each result is obtained from the average of ten iterations for each classifiers. Data for the first 4 case studies indicate that all FRBCs have good generalization capabilities since the training and testing accuracies are very close. In the last case study, the perturbation of the test data set has inevitably decreased the testing accuracy. Comparing the first and the second case studies, the performance differences between the mean calculation method of simple averaging and FCM are minimal. This shows the FCM does not bring any improvement over the simple averaging method. In all cases, T2 FRBCs consistently outperform BL-T1

FRBC, this shows that the FOU is essential for a better FRBC. In particular, the third and the fourth case studies show that T2 classifiers are more robust against the perturbations in the MFs, hence less sensitive to design errors. BL-T1 classifier has significant testing performance drop ($\approx 5\%$) when the test data are corrupted by noises as shown in last case study while all T2 FRBCs remain relatively robust with only $1-2\%$ performance drop. This implies that the issue of unpredictability can be minimized through the proposed type-2 framework. The boxplots are shown in Fig. 15.8-15.12. It is clear that under any types of the perturbations mentioned above, T2 FRBCs remain robust and consistent compared to T1 FRBC.

15.6 Conclusion

In this chapter, we presented some simple yet intuitive fuzzy approaches to ECG arrhythmic classification namely to distinguish NSR, VF and VT. The structure of the FRBC and the initial parameters can be computed directly from the data set. When dealing with biological signals such as ECG, various uncertainties can arise and this will lead to the usability limitation of the algorithm in real-world applications. Through the extensive experimental results, we have shown that T2 FRBCs has much better and stable performances in face of different sources of uncertainties. The optimization of the FOUs by using GA is proven to be effective and model-free. While there is no preference for T2-US, T2-UM or T2-USUM FRBCs in this application, there is always a possibility that either one of them will outperform the rests in other applications. Finally, we believe that the FRBC design strategy described in this chapter provide the general methodology that can be also be applied to other classification tasks.

References

1. Bellman, R.E., Kalaba, R., Zadeh, L.A.: Abstraction and pattern classification. J. Math. Anal. Appl. 13, 1–7 (1966)
2. Chua, T.W., Tan, W.W.: Ga optimisation of non-singleton fuzzy logic system for ecg classification. In: IEEE Congress on Evolutionary Computation, pp. 1677–1684 (2007)
3. Kerber, R.E., et al.: Automatic external defibrillators for public access defibrillation: Recommendations for specifying and reporting arrhythmia analysis, algorithm performance, incorporating new waveforms, and enhancing safety. Circulation 95, 1677–1682 (1997)
4. Hwang, C., Rhee, F.: Uncertain fuzzy clustering: Interval type-2 fuzzy approach to c-means. IEEE Trans. Fuzzy Systems 15(1), 107–120 (2007)
5. Mendel, J.M.: Uncertain rule-based fuzzy logic systems: Introduction and new directions. Prentice-Hall, Englewood Cliffs (2001)
6. Mendel, J.M., John, R.I.: Type-2 fuzzy sets made simple. IEEE Transactions on Fuzzy Systems 10(2), 117–127 (2002)
7. Mendela, J.M., Hagras, H., John, R.I.: Standard background material about interval type-2 fuzzy logic systems that can be used by all authors. IEEE Computational Intelligence Society

8. Mit/bih database distribution, Massachusetts Inst. Technol., Cambridge, MA
9. Wolkenhauer, O.: Possibility theory with applications to data analysis. Research Studies Press, Tauton (1998)
10. Wolkenhauer, O.: Data engineering: Fuzzy mathematics in systems theory and data analysis. John Wiley and Sons, New York (2001)
11. Zhu, Y.S., Zhang, X.S., Thakor, N.V.: Detecting ventricular tachycardia and fibrillation by complexity measure. IEEE Transactions on Biomedical Engineering 46(5), 837–843 (1990)

16

Fuzzy Logic in Evolving *in silico* Oscillatory Dynamics for Gene Regulatory Networks

Yaochu Jin and Bernhard Sendhoff

Honda Research Institute Europe, Carl-Legien-Str. 30, 63073 Offenbach, Germany
{yaochu.jin,bernhard.sendhoff}honda-ri.de

Summary. This chapter investigates empirically the influence of control logic on regulatory dynamics in computational models of genetic regulatory networks. The gene regulatory network motif considered in this work consists of three genes with both positive and negative feedback loops. Two fuzzy logic formulations are studied in this work, one is known as the Zadeh operator, and other is the probabilistic operator. The evolved dynamics of the network motifs is then verified with perturbed initial system states. Our empirical results show that with the probabilistic 'AND' operator and the probabilistic 'OR' operator, the system is able to evolve sustained oscillation with a low probability. However, sustained oscillation is not evolvable when the Zadeh operator is employed. In addition, we also show that regulatory motifs with the probabilistic operators possess much richer dynamics than that with the Zadeh operators.

16.1 Introduction

Modeling and analysis of gene regulatory networks is receiving increasing attention in computational systems biology. It has been found that a small number of sub-networks, also known as network motifs, occur very often in complex gene regulatory networks. These network motifs serve as building blocks of regulatory networks and the dynamics of the whole networks can be analyzed by analyzing these motifs. Detection and analysis of regulatory motifs in biological systems has now become one important research topic in systems biology [1, 2].

One line of fascinating research is to analyze the role of positive and negative feedback loops in the robustness and evolvability of gene regulatory networks. It has been found that negative feedback loops are a major mechanism for biological robustness, e.g., in heat shock response of *E. Coli* [6] and in perfect adaptation of bacteria chemotaxis [20]. A design principle found in cell signaling networks is that coherently coupled feedback loops are of essential importance to robustness [13], and that networks containing a large number of positive feedback loops and a small number of negative feedback loops are more likely to be robust to perturbations [14]. Most recently, it has been reported that a combination of positive feedback with negative feedback loops endows the networks with more robust and tunable sustained oscillations, and makes it easier to evolve stable oscillatory dynamics [19].

Y. Jin and L. Wang (Eds.): Fuzzy Systems in Bio., STUDFUZZ 242, pp. 315–327.
springerlink.com

Meanwhile, regulatory control, particularly the regulation logic, is also attracting more and more research efforts. An experimental analysis of regulation control of the gene for development of the sea urchin has been conducted in [21]. A systematic investigation of control logic in gene regulation has been performed in [17], which concludes that, among others, networks consisting of competitively binding activators and repressors can be controlled more robustly. Another interesting finding suggests that two types of logic control may exist in bacteria transcriptional networks, namely, a digital type and analog type [15]. Interestingly, these two distinct control types are found to be complementary in gene regulation. Negative feedback loops that promote systems robustness to mutations have also been shown to emerge in computational evolution of developmental system [18].

This chapter investigates *in silico* the role of regulation logic in evolving oscillatory dynamics for a regulatory motif consisting of a negative feedback loop and a positive feedback loop. To this end, we employ an evolution strategy, one of the widely used artificial evolutionary algorithms [3], to evolve the parameters of the given network motifs. Though evolution of the desired dynamics for a given network motif appears straightforward at the first sight, we find out that it is nontrivial to evolve sustained oscillations, i.e., limit cycles. The most interesting finding from this work is that the probabilistic fuzzy operators can produce much richer dynamics than their Zadeh counterparts, and thus have a better ability to evolve sustained oscillatory dynamics.

A few research efforts have been made to evolve dynamics for gene regulatory networks *in silico*. In [7], both bistable switches and oscillators are evolved based on a number of predefined basic biochemical reactions. However, it was suggested in [4] that the results reported in [7] are not easily reproducible, which implies that successful evolution of sustained oscillation is sensitive to experimental setups. In [4], a correlation based fitness function has been suggested, though no definite conclusion can be drawn on its influence on the successful evolution of oscillators. Similar work has also been reported in [16], where two different fitness functions are suggested for evolving oscillation. In [11], it is shown that a higher Hill co-efficient facilitates the evolution of sustained oscillation for the relaxation oscillator.

This chapter is organized as follows. In Section 16.2, a brief introduction to gene expression and the mathematical model of the studied network motif is provided. The concept of regulation control is discussed in Section 16.3, where a number of fuzzy logic expressions is also presented. Section 16.4 describes very briefly the evolution strategy used in this work. Experimental results are given in Section 16.5 with discussions, and Section 16.6 concludes this chapter.

16.2 Regulatory Network Motifs and Regulation Control Logic

According to the central dogma of biology, the process of gene expression is believed to be composed of two main steps, namely, transcription of DNA to

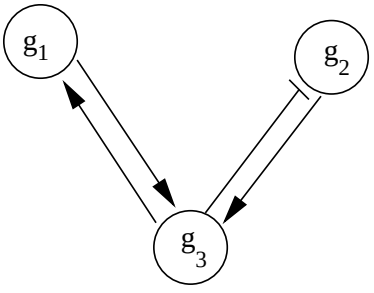


Fig. 16.1. A regulatory motif consisting of three gene, forming a negative feedback loop and a positive feedback loop

mRNA and translation of the mRNA to encoded proteins. The expression of genes is controlled by biophysical and biochemical interactions among genes, proteins and metabolites. This network of interactions is termed the gene regulatory network.

The regulatory motif studied in this work consists of three genes, as shown in Fig. 16.1. From the figure, we can see that genes g_1 and g_3 formulate a positive feedback loop, while genes g_2 and g_3 build up a negative feedback loop. The mathematical model of the regulatory motif can be described by the following differential equations:

$$\dot{x}_1 = a_{13}H_{13}(x_3) - a_{11}x_1, \tag{16.1}$$

$$\dot{x}_2 = a_{23}H_{23}(x_3) - a_{22}x_2, \tag{16.2}$$

$$\dot{x}_3 = a_3 L(H_{31}(x_1), H_{32}(x_2)) - a_{33}x_3, \tag{16.3}$$

where $x_i, i = 1, 2, 3$ are the concentration of the corresponding protein products of the three genes, a_{11}, a_{22}, and a_{33} are the degradation rate of the proteins, and a_{13}, a_{23}, and a_3 are the parameters representing the strength of the protein interactions. All these parameters are non-negative, and

$$H_{13}(x_3) = \frac{\beta x_3^n}{\theta_1^n + x_3^n}, \tag{16.4}$$

$$H_{31}(x_1) = \frac{\beta x_1^n}{\theta_3^n + x_1^n}, \tag{16.5}$$

$$H_{32}(x_2) = \frac{\beta x_2^n}{\theta_4^n + x_2^n} \tag{16.6}$$

$$H_{23}(x_3) = \frac{\beta}{1 + (x_3/\theta_2)^n}, \tag{16.7}$$

where β, $\theta_i, i = 1, 2, 3, 4$, and n are parameters in the activating and repressive Hill functions, where n is called the Hill coefficient. We can see that H_{13}, H_{31} and H_{32} are activating, and H_{23} is repressive.

In Equation (16.3), $L(H_{31}, H_{32})$ is the function denoting the regulation logic that combines the influence of activating regulations from g_1 and g_2 of the expression of g_3. Often the case, various regulatory inputs are supposed to be additive. However, this may not be always true in biology, as discussed in [17]. Generally, activating interactions from g_1 and g_2 can be either independent, competitive or cooperative. In this work, we consider two situations. 1) Both transcription factors produced by g_1 and g_2 are necessary for the expression of g_3, and 2) Either of the transcription factors will be sufficient for the expression of g_3. These two situations can be described by logic 'AND', logic 'OR', respectively. In the following section, we are going to introduce in more details the logic functions used in this work.

16.3 Fuzzy Logic

Fuzzy logic systems have found a wide range of applications in science and engineering [12] since Zadeh's pioneering work on fuzzy sets [22] and fuzzy reasoning [23]. It is believed that fuzzy logic systems are particularly powerful in dealing with uncertainties in modeling, reasoning, and control, just to name a few. The unique ability of fuzzy systems can be attributed, in part, to the following two features. First, in contrast to the conventional set theory, where an element either belongs to or does not belong to a set, while in the fuzzy set theory, an element may belong to a set with a degree between zero and one. This membership degree is defined by a piece-wise continuous membership function whose value is between 0 and 1. Second, the fuzzy logic operators that allow for more flexible processing of information. In the earlier age, most fuzzy systems were built upon human heuristics, or from observations of human experts. Since the beginning of 1990's, data-driven fuzzy systems have been playing an increasingly important role, where fuzzy rules are abstracted from experimental data. One main new feature of the data-driven fuzzy systems is their ability to learn, which can be largely attributed to the marriage of machine learning techniques, such as neural networks and evolutionary algorithms with fuzzy set theory [5, 8, 10]. However, it must be pointed out that data-driven fuzzy systems may lose interpretability, which is the essence of fuzzy systems [10]. To address this problem, interpretability issues should be taken into account [9] in generating fuzzy rules from data.

This work investigates the role of fuzzy logic in modeling gene regulation control and its relationship to evolving oscillatory dynamics. In gene regulation, the expression of a gene is often regulated by a number of regulatory units (enhancers or silencers), and more than one transcription factor can be bound to a binding site. The question is, does it need all activating TFs to activate the expression of the gene, or is it sufficient to have only one of the TF to activate? Another question is, if multiple TFs can be bound to the binding site, are they independent, competitive, or as a compound only?

In this work, we try to answer the first question mentioned above to a certain degree. As shown in Fig. 16.1, the regulatory motif we are studying contains three

genes, where gene 3 is activated by both gene 1 and gene 2. Two possibilities are considered in the following: 1) Genes 1 and 2 are both needed to activate gene 3, which can be modeled using the fuzzy 'AND' logic; 2) Either gene 1 or gene 2 are needed to activate gene 3, modeled using the fuzzy 'OR' logic.

Two types of fuzzy logic formulations are investigated in the simulations. The first type is the Zadeh operators:

$$\text{AND: } x \bigwedge y = \min(x, y), \tag{16.8}$$

$$\text{OR: } x \bigvee y = \max(x, y), \tag{16.9}$$

where '$\bigwedge$' and '$\bigvee$' denote fuzzy 'AND' and 'OR', respectively, $\min(x, y)$ and $\max(x, y)$ return the minimum and the maximum of x and y, respectively.

The second type of fuzzy logic operators is known as the probabilistic operators, which can be described as follows:

$$\text{AND: } x \bigwedge y = xy, \tag{16.10}$$

$$\text{OR: } x \bigvee y = x + y - xy. \tag{16.11}$$

It should be mentioned that in logic operations, the value of x and y is always limited between zero and one. In this work, we require that the value is non-negative, but it is allowed to be larger than one.

16.4 Evolution Strategy

Evolution strategies are one of the widely used artificial evolutionary algorithms that are very effective for optimizing real-valued problems. Since the structure of the regulatory motifs is fixed, and only the parameters are evolved in this work, we adopt a canonical evolution strategy for evolving the desired dynamics. In a canonical evolution strategy (ES), the mutation of the object parameters (the parameters to be optimized) is performed by adding an $N(0, \sigma_i^2)$ distributed random number, where σ_i's are termed as strategy parameters that are also encoded in the genotype and subject to mutations. The ES used in this work can be described as follows:

$$\mathbf{x}(t) = \mathbf{x}(t-1) + \tilde{\mathbf{z}} \tag{16.12}$$

$$\sigma_i(t) = \sigma_i(t-1)\exp(\tau' z)\exp(\tau z_i); i = 1, ..., n, \tag{16.13}$$

where $\mathbf{x}$ is an n-dimensional parameter vector to be evolved, $\tilde{\mathbf{z}}$ is an n-dimensional random number vector with $\tilde{\mathbf{z}} \sim N(\mathbf{0}, \sigma(t)^2)$, z and z_i are normally distributed random numbers with $z, z_i \sim N(0, 1)$. Parameters τ, τ' and σ_i are the strategy parameters, also known as step-sizes, where σ_i is mutated as in equation (16.13) and τ, τ' are constants as follows:

$$\tau = \left(\sqrt{2\sqrt{n}}\right)^{-1}; \ \tau' = \left(\sqrt{2n}\right)^{-1}. \tag{16.14}$$

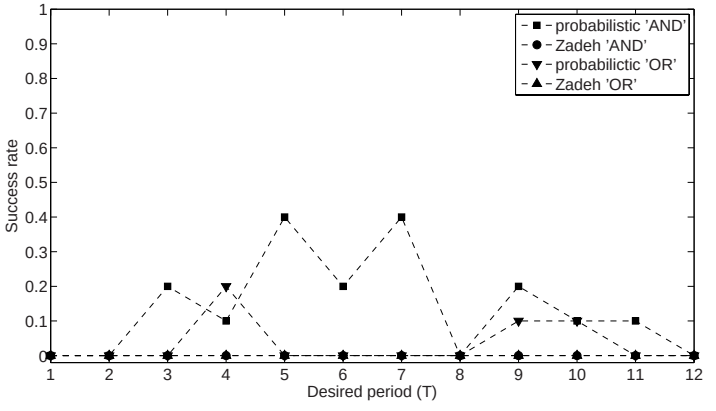


Fig. 16.2. Success rate in evolving sustained oscillation

Two selection schemes have been proposed in evolution strategies, known as comma and plus strategies. Suppose there are μ and λ individuals in the parent and offspring population, usually $\mu \leq \lambda$. In the comma strategy, μ parent individuals are selected only from the λ offspring individuals, which is usually noted as (μ,λ)-ES. In the plus strategy, μ parent individuals are selected from a union of μ parent individuals and λ offspring individuals, which is noted as $(\mu+\lambda)$-ES. In our study, the (μ, λ)-ES is adopted.

In the evolution, all parameters in the regulatory model, i.e., three decay rates and three synthesis rates, one coefficient (β), four thresholds, and one Hill coefficient are the object parameters encoded in the genome.

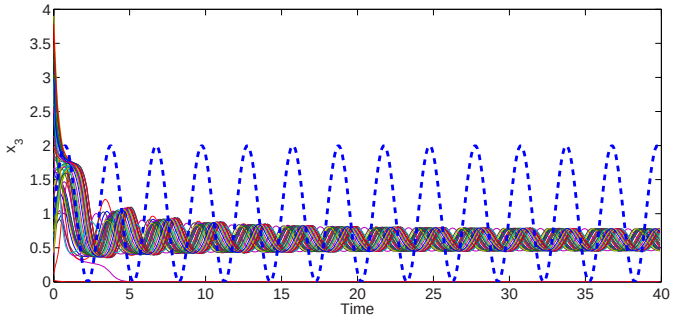
16.5 Simulation Results

A (30, 200)-ES has been adopted in our experiments. All object parameters parameters to be evolved are randomly initialized between 0 and 4. According to the physical meaning of the parameters, a lower bound is set to 0 for all parameters, but no upper bound is given. The initial step-size is set to 1. In all simulations, 500 generations are run for each case.

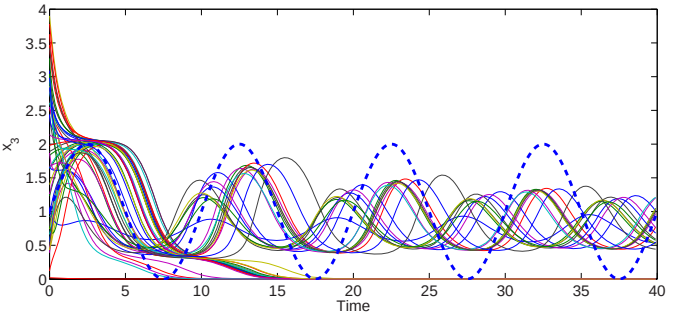
The aim of the work is to produce a sustained oscillatory dynamics for the concentration of g_3. The target function for x_3 in evolving oscillation is defined by a sinus function as follows:

$$x_3^d(t) = \sin(2\,\pi\,t/\mathrm{T}) + 1.0, \tag{16.15}$$

where t is time instant, and T is the desired period of the oscillation. In the simulations, a desired period of $T = 1, 2, ..., 12$ is chosen in 12 groups of simulations, and for each desired period, 10 independent runs are performed. During the evolution, x_1 and x_2 are initialized to 1.0, while x_3 is initialized to 0.



(a)



(b)

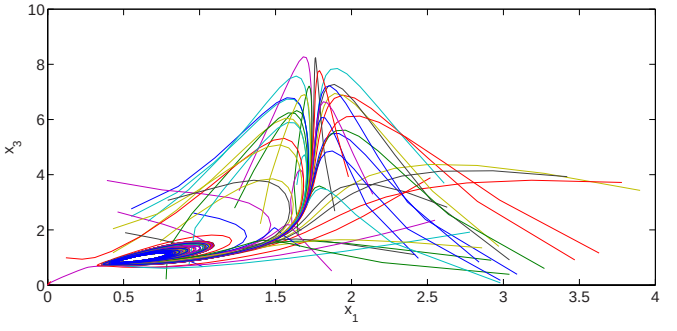
Fig. 16.3. Time course of x_3 of the evolved gene regulatory network given 50 randomly initialized states. The thick dashed line denotes the desired signal. (a) $T = 3$, and (b) $T = 10$.

16.5.1 Easiness of Evolving Sustained Oscillation

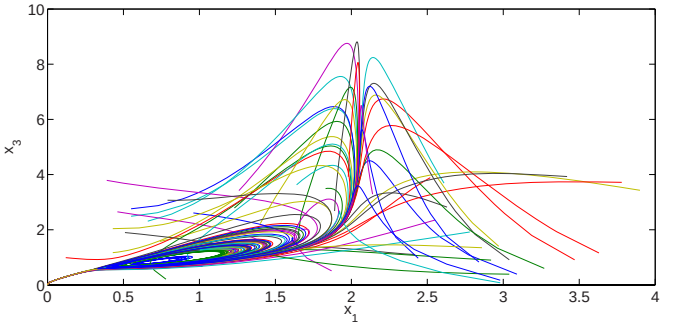
Fig. 16.2 shows the rate of successful evolution of sustained oscillation in 10 independent runs, when different logic functions are employed. It can be seen that when the probabilistic 'AND' operator is used, the system is able to evolve sustained oscillation for 17 times from a total of 120 independent runs. When the probabilistic 'OR' is used, sustained oscillation is evolved only four times in 120 runs.

Although the successful rates are quite low in general, it is interesting to notice that a large Hill coefficient is not required, as suggested in [11], where different regulatory signals are summed up in a relaxation oscillator.

Apart from the different success rates for different logic formulations, evolved regulatory motifs with probabilistic logic operators show richer dynamics than



(a)



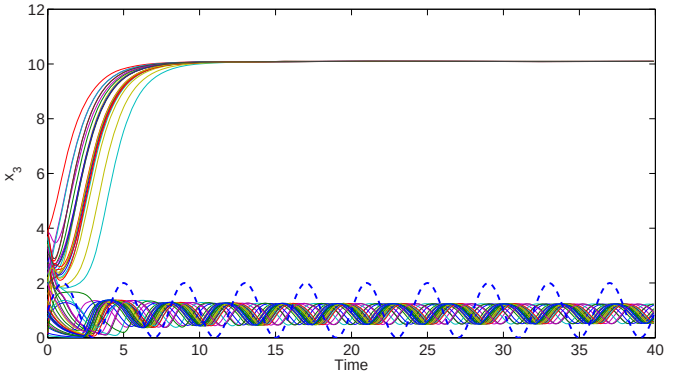
(b)

Fig. 16.4. State-space trajectories of the evolved regulatory motifs. In both cases, the system has a limit cycle and a stable equilibrium. (a) $T = 3$, and (b) $T = 10$.

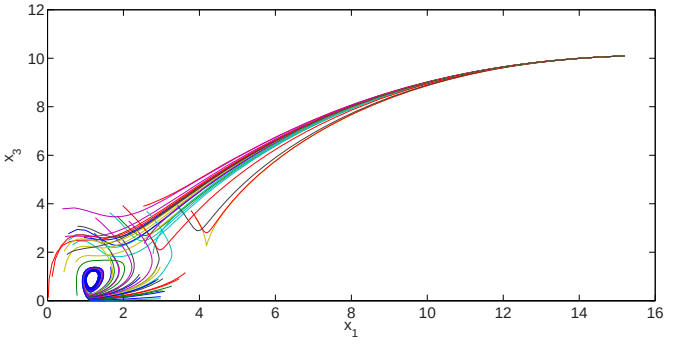
those with Zadeh operators in that the former often consists of two different dynamic features, usually a limit cycle plus a stable equilibrium or an attractor plus a stable equilibrium. In the latter case, most systems generate either an attractor or a stable equilibrium only.

16.5.2 Analysis of Evolved Dynamics

In this subsection, we examine the dynamics of the evolved network motifs, both for cases where sustained oscillation has or has not been successfully evolved. At first, we check two runs using the probabilistic 'AND' with successfully evolved oscillation, where the desired period of the oscillation is 3 and 10, respectively. The resulting dynamics are plotted in Fig. 16.3, where the initial states of $x_i(0), i = 1, 2, 3$ are randomly set between 0.0 and 4.0. We find that for $T = 3$, refer to Fig. 16.3(a), 48 of the 50 initial states result in sustained oscillation, while the other two initial states converge to 0. In contrast, 19 initial states



(a)

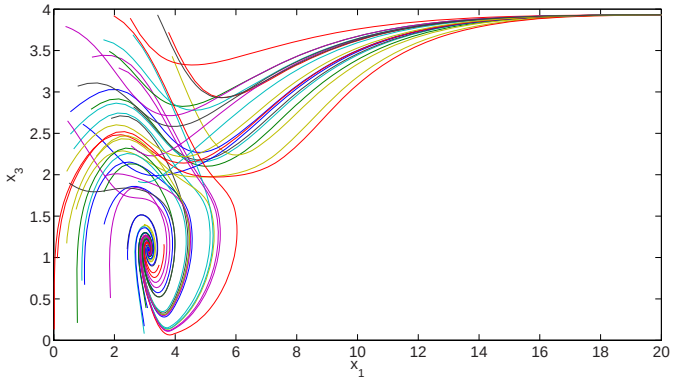


(b)

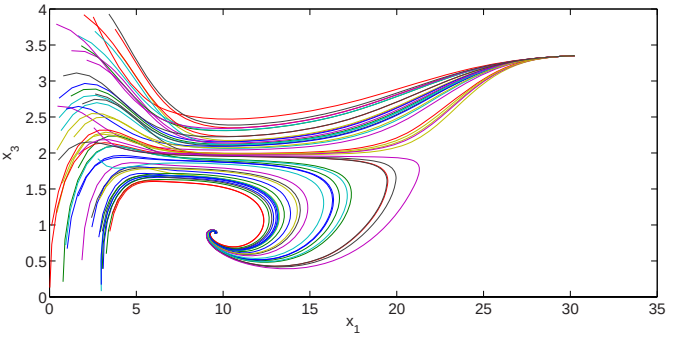
Fig. 16.5. (a) Time course of x_3 of the evolved regulatory motif when the probabilistic 'OR' is used (the desired period $T = 4$). (a) Time course, and (b) state-space trajectory of 50 random initial states.

result in sustained oscillation, while the rest converge to zero, for $T = 10$, see Fig. 16.3(b). In addition, we can make the following observations. First, the frequency of the evolved oscillator is roughly the same as the desired frequency, as can be seen in Fig. 16.3. Second, the amplitude of the limit cycles for different initial states are very similar. Third, both evolved motifs have one limit cycle plus one equilibrium, refer to Fig. 16.4.

In the following, we are going to present some additional interesting dynamics observed in the evolved regulatory motifs.



(a)



(b)

Fig. 16.6. Bi-stabile dynamics from an evolved regulatory motif with probabilistic 'OR' logic. (a) $T = 6$, and (b) $T = 12$.

Limit Cycle together with a Stable Equilibrium

A limit cycle plus an equilibrium has been evolved when a probabilistic 'AND' (refer to Fig. 16.4) or a probabilistic 'OR' is used, see Fig. 16.5. In the former case, the system states go very closely by the limit cycle and then converge to the origin. In the latter case, the location of the limit cycle and the equilibrium is quite distant from each other.

Bi-stability

Although the desired dynamics is a sustained oscillation, bi-stable dynamics often emerges in the evolved systems, particularly when the probabilistic 'OR' logic is used. Two kinds of bi-stable dynamics have been generated, either

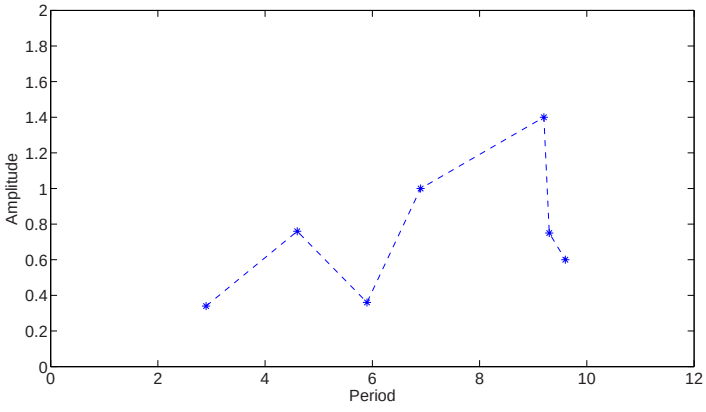


Fig. 16.7. Amplitude and period of the evolved limit cycles

one oscillatory attractor and an equilibrium as shown in Fig. 16.6(a), or two equilibria, see Fig. 16.6(b).

Frequency-Amplitude Relation

It has been observed that it is of great importance in biology that a regulatory system can show adaptive oscillation frequencies with a similar oscillation amplitude [19]. Since the motif we studied in this work also contains a negative feedback loop plus a positive one, we try to verify this observation when the probabilistic 'AND' logic is used. The relationship between the evolved period and the amplitude of the limit cycles are presented in Fig. 16.7. In these network motifs, the amplitude ranges from 0.4 to 1.4 when the period varies from 2.9 to 9.3.

16.5.3 Discussions

The biological meanings of the dynamics evolved for the regulatory motifs in this work remains to be revealed. On the one hand, we show that rich dynamics, such as limit cycles, attractors, equilibria, as well as bistability, has been evolved successfully for a very simple regulatory motif when the probabilistic logic operators are used for modeling the regulation logic, which signifies that the systems ability to generate rich phenotypic features. On the other hand, the biological implication of such richness in dynamics, particularly its role in biological evolution, is still unclear.

16.6 Conclusions

This chapter reports our initial results on the influence of fuzzy regulation logic on the easiness of evolving oscillatory dynamics for a gene regulatory motif. It has been shown that the probabilistic fuzzy logic can produce richer dynamics than

the Zadeh operators. Three interesting phenomena have been observed. First, a range of desired frequencies can be evolved in case the probabilistic 'AND' logic is used. For probabilistic 'OR' logic, the probability to evolve sustained oscillation is much lower. When the Zadeh 'AND' or Zadeh' OR operation is used, no sustained oscillation can be evolved in a total of 120 independent runs performed in this work. Second, the frequency and amplitude of the generated oscillation is quite robust to the initial states of the system, and the difference of the amplitude for different frequencies is relatively small. Third, the attraction basin of a sustained, or dampened oscillation covers only part of the state-space, and usually, there is also an equilibrium in addition to the limit cycle. Fourth, although a sustained oscillation is targeted, the resulting dynamics can also be bi-stability, or equilibria only.

A few interesting issues remain to be investigated. For example, how the evolvability of sustained oscillation may change as the activation pattern evolved. In this work, the target gene has two activating regulatory connections, with one positive feedback loop and one negative feedback loop. It is of interest to check if coherency of the feedback loops and the coherency of the transcription factors also play a role in the evolvability.

Another limitation of this work is that only the parameters are changed with a predefined wiring structure and fixed regulation logic. It is of interest to study if an evolvable structure in terms of both regulatory structure and regulation logic can be evolved autonomously.

References

1. Alon, U.: An introduction to systems biology: Design principles of biological circuits. Chapman & Hall, Boca Raton (2007)
2. Alon, U.: Network motifs: theory and experimental approaches. Nature Review Genetics 8, 450–461 (2007)
3. Bäck, T.: Evolutionary algorithms in theory and practice: Evolution strategies, evolutionary programming, genetic algorithms. Oxford University Press, Oxford (1996)
4. Chu, D.: Evolving genetic regulatory networks for systems biology. In: Congress on Evolutionary Computation, pp. 875–882. IEEE Press, Los Alamitos (2007)
5. Cordon, O., Herrera, F., Hoffmann, F., Magdalena, L.: Genetic fuzzy systems - evolutionary tuning and learning of fuzzy knowledge bases. World Scientific, Singapore (2001)
6. El-Samad, H., Kurata, H., Doyle, J.C., Gross, C.A., Khammash, M.: Surviving heat shock: Control strategies for robustness and performance. Proceedings of the National Academy of Science of the USA 102(8), 2736–2741 (2005)
7. Francois, P., Hakim, V.: Design of genetic networks with specified functions by evolution in silico. Proceedings of the National Academy of Sciences of USA 101(2), 580–585 (2007)
8. Jang, J.-S.R., Lin, C.-T., Mitzutani, E.: Neuro-fuzzy and soft computing. Prentice Hall, Englewood Cliffs (1997)
9. Jin, Y.: Fuzzy modeling of high-dimensional systems: Complexity reduction and interpretability improvement. IEEE Transactions on Fuzzy Systems 8(2), 212–221 (2000)

10. Jin, Y.: Advanced fuzzy systems design and applications. Physica/Springer, Heidelberg (2003)
11. Jin, Y., Sendhoff, B.: Evolving in silico bistable and oscillatory dynamics for gene regulatory network motifs. In: Congress on Evolutionary Computation, pp. 386–391 (2008)
12. Klir, G.J., Clair, U.S., Yuan, B.: Fuzzy set theory: Foundations and applications. Prentice-Hall, Englewood Cliffs (1997)
13. Kwon, Y.-K., Cho, K.-H.: Co-herent coupling of feedback loops: A design principle of cell signalling networks. Bioinformatics 24(17), 1926–1932 (2008)
14. Kwon, Y.-K., Cho, K.-H.: Quantitative analysis of robustness and fragility in biological networks based on feedback dynamics. Bioinformatics 24(7), 987–994 (2008)
15. Marr, C., Geertz, M., Hütt, M.-T., Muskhelishvili, G.: Dissecting the logic of network control in gene expression profiles. BMC Systems Biology 2, 18 (2008)
16. Paladugu, S.R., Chickarmane, V., Dekard, A., Frumkin, J.P., MaCormarck, M., Sauro, H.M.: In silico evolution of functional modules in biochemical networks. IEEE Proceedings on Systems Biology 153(4), 223–235 (2006)
17. Schilstra, M.J., Nehaniv, C.L.: Bio-logic: Gene expression and the laws of combinatorial logic. Artificial Life 14, 121–133 (2008)
18. Steiner, T., Schramm, L., Jin, Y., Sendhoff, B.: Emergence of feedback in artificial gene regulatory networks. In: Congress on Evolutionary Computation, pp. 867–874 (2007)
19. Tsai, T.Y.-C., Choi, Y.S., Ma, W., Pomerening, J.R., Tang, C., Ferrel Jr., J.E.: Robust, tunable biological oscillations from interlinked positive and negative feedback loops. Science 321, 126–129 (2008)
20. Yi, T.-M., Huang, Y., Simon, M.I., Doyle, J.: Robust perfect adaptation in bacterial chemotaxis through integral feedback control. Proceedings of the National Academy of Science of the USA 97(9), 4649–4653 (2000)
21. Yuh, C.-H., Bolouri, H., Bower, J.M., Davidson, E.H.: A logical model of cis-regulatory control in a Eukatyotic system. In: Bower, J.M., Bolouri, H. (eds.) Computational Modeling of Genetic and Biochemical Networks, pp. 73–100. MIT Press, Cambridge (2000)
22. Zadeh, L.A.: Fuzzy sets. Information Control 8(3), 338–353 (1965)
23. Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and decision processes. IEEE Transactions on Systems, Man, and Cybernetics 3(1), 28–44 (1971)

Index

Author Index